ENHANCED UNIFORM MANIFOLD APPROXIMATION AND PROJECTION VIA SIMULTANEOUS PERTURBATION STOCHASTIC APPROXIMATION

by
Adam Byerly

A thesis submitted to Johns Hopkins University in conformity with the requirements for the degree of Master of Science.

Baltimore, Maryland
May 2021

Abstract

Traditional machine learning and statistical analysis techniques often breakdown when applied to high dimensional data. While significant progress has been made in processing such data [1, 2], these techniques often attempt to exploit some sub-structure prevalent to the data. Dimensionality reduction is a broad class of techniques designed to specifically reduce the dimensionality of data, while preserving relevant structure for further processing, e.g. clustering. Uniform Manifold Approximation and Projection (UMAP) [3] is a state-of-the-art non-linear dimension reduction algorithm that constructs a topologically motivated graph representation of the data, before optimizing the low-dimensional representation of this graph. Current implementations of UMAP use a stochastic gradient estimate of a constructed smooth approximation while performing this optimization. The Simultaneous Perturbation Stochastic Approximation (SPSA) [4] algorithm, which only requires two measurements of the loss function when computing gradient estimates, bypasses the need for this smooth approximation. This thesis introduces the UMAP-SPSA algorithm to perform the UMAP dimension reduction without the need for the smooth approximator. Further, we analyze the the algorithm's computational performance and embedding accuracy.

Thesis Readers

Dr. Stacy Hill (Primary Advisor)
Senior Professional Staff
Johns Hopkins University Applied Physics Laboratory

and

Research Faculty Member
Applied and Computational Mathematics
Johns Hopkins Whiting School of Engineering

Dr. Thomas Woolf (Second Reader)
Research Faculty Member
Applied and Computational Mathematics
Johns Hopkins Whiting School of Engineering

Acknowledgements

I would like to thank my advisor Dr. Stacy Hill for his support of the work presented here. This would not have been possible without the guidance he provided throughout this research and my Masters journey at large.

I would like to thank Dr. Thomas Woolf for his passion in teaching and for his mathematical insights.

I would also like to thank all of my other professors and the Department of Applied and Computational Mathematics for helping me get this far.

I would like to thank all my colleagues, friends, and family for their support along the way.

Lastly, I am forever grateful for my wife, for without her love and support, I would never be where I am today.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Recent advances in technology have driven a massive increase in the volume, velocity, and dimension of data available for measurement and collection [5]. Many diverse fields have benefited from this proliferation, such as neurology, sensor fusion, and finance [6, 7, 8]. While learning techniques exist for analysis and prediction in these fields, they typically break down in high dimensions due to the *curse of dimensionality* [9, 10]. If one were able to reduce the dimensionality of such large datasets while preserving the structure of interest, then these classes of techniques would again become viable.

1.2 Dimension Reduction

Dimension reduction is the process of finding lower-dimensional representations of high-dimensional data, while preserving some relevant structure, e.g., sample distance. The study of dimension reduction techniques has its roots in Pearson's seminal paper on lines of best fit [11]. Pearson was originally interested in finding curves and surfaces of best-fit given a set of (potentially noisy) measurements. In Pearson's case, the data of interest was measured in some ambient space (each dimension corresponding to a measured property), but the relationship of interest was found when projecting the data to these lower-dimensional lines and planes.

Formally, the general dimension reduction problem seeks to find a function

$$f : \mathbf{A} \to \mathbf{E}, \tag{1.1}$$

where $\mathbf{A}$ is the ambient data space and $\mathbf{E}$ is the embedded data space, with $\dim(\mathbf{A}) < \dim(\mathbf{E})$. Such an f need not be injective, for if we consider the canonical projections $\pi_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \cdots, n$ such that $\pi_i(x_1, \cdots, x_n) = x_i$, then f is clearly non-injective.

The linearity of f divides techniques into two broad families, namely linear and non-linear. That is, if f is a linear mapping, i.e., $f(x + y) = f(x) + f(y)$ and $f(\alpha x) = \alpha f(x)$, then the dimension reduction technique is a linear technique; otherwise it is non-linear. Linear methods, such as Principle Component Analysis (PCA) [11], Factor Analysis [12], and Linear Discriminant Analysis (LDA) [13] have long been used to perform dimension reduction, but such approaches typically fail to appropriately account for nonlinearities that can arise in complex sets of data. Recent work has instead explored non-linear techniques for reducing the dimension of complicated data.

In particular, a specific sub-class of non-linear dimension reduction has been of interest recently which focuses on constructing a neighbor-graph representation of the dataset of interest, then seeks to efficiently embed this graphical representation. Techniques such as Laplacien Eigenmaps [14], t-Distributed Stochastic Neighbor Embedding (t-SNE) [15], and Uniform Manifold Approximation and Projection [3] have all adopted this approach, and recently UMAP has achieved state-of-the-art performance in terms of computational runtime and embedding accuracy across many open machine learning datasets. In our study, we explore a modification to the UMAP algorithm which we feel enhances the embedding accuracy while maintaining a computationally competitive runtime.

1.3 Summary of Contents

The remainder of this thesis is as follows: in Chapter 2 we summarily review the Laplacian Eigenmaps (LE) algorithm as a precursor for Uniform Manifold Approximation and Projection (UMAP), before presenting the UMAP and Simultaneous Perturbation Stochastic Approximation (SPSA) algorithms in detail. Then in Chapter 3, we introduce our UMAP modification to leverage SPSA optimization. In Chapter 4 we provide numerical results on the quantitative and computational performance of the algorithm as compared to other dimension reduction techniques. Finally, we conclude this thesis in Chapter 5 by discussing the strengths and weaknesses of our approach, and suggest possible research directions for future study.

Chapter 2

Preliminaries

2.1 The Nerve Theorem

While we focus on the Uniform Manifold Approximation and Projection (UMAP) algorithm in this work, and to an extent its spectral precursor Laplacian Eigenmaps (LE), the underlying neighbor-graph construction technique is common to many dimension reduction techniques. Several non-linear dimension reduction techniques make the same initial assumption: suppose $X = \{x_1, \cdots, x_n\}$ is a set of points lying on a manifold $\mathscr{M}$ embedded in ambient $\mathbb{R}^D$ space. This manifold $\mathscr{M}$ is of a lower-dimension than $\mathbb{R}^D$, but it is not observed; only samples from this manifold are available (the observed X). From this discrete subset X, we seek to recover the manifold structure of $\mathscr{M}$.

Fortunately, the machinery to recover such structure is possible by way of the **Čech complex**. Given a set of points X, the Čech complex $\check{C}_\epsilon(X)$ is built as follows: for each subset of X, $\sigma \subset X$, $\sigma \in \check{C}_\epsilon(X)$ if and only if the intersection of every ϵ-ball centered at each $\sigma_i \in \sigma$ is non-empty. Such a construction is important because it has been shown that $\check{C}_\epsilon(X)$ is homotopy-equivalent (see Appendix A) to the union of ϵ-ballscentered at each point in X, and thus the manifold structure local to the sampled points may be recovered [16]. This is known as the **nerve theorem**. Niyogi, Smale, and Weinberger [17] further extend this result to encompass cases where the sampled X is “noisy” and lies near rather than on the manifold in question. By constructing a neighbor-graph of the sampled points X in the "right" way, one can now exploit the nerve theorem when constructing a dimension-reducing embedding by recovering the underlying manifold structure. For a detailed review on the underlying Homology theory, we refer the reader to [18], [19], or [16].

2.2 Laplacian Eigenmaps

We present here the details of the Laplacian Eigenmaps (LE) algorithm, an important precursor to UMAP. Given a set of points $X = \{x_1, \cdots, x_n\}$ lying on a manifold $\mathscr{M}$ embedded in $\mathbb{R}^D$, the LE algorithm seeks to find an embedding $f : X \to \mathbb{R}^E$ with $E \ll D$. The algorithm constructs such an f in two key phases: construction of a weighted adjacency graph, and construction of the eigenmap embedding.

The LE algorithm considers the ambient data X as an adjacency graph's vertices, i.e. $V = X$, then connects each vertex x_i to its nearest k neighbors

$$e_{ij} = \begin{cases} 1, & x_j \in N_k(x_i), \\ 0, & \text{otherwise}, \end{cases} \tag{2.1}$$

where $N_k(x_i)$ is the set of k-nearest neighbors to x_i as defined by $d_{\mathbb{R}^D}$ (the metric on $\mathbb{R}^D$), and k is an algorithmic *hyperparameter*. Choice of k is motivated by how much local, versus global, neighborhood information is to be encoded in the adjacency graph. For each edge, e_{ij} from x_i to x_j, the algorithm then defines the edge weighting as

$$w_{ij} = \begin{cases} \exp\left(-\frac{\|x_i - x_j\|^2}{t}\right), & e_{ij} = 1, \\ 0, & \text{otherwise}, \end{cases} \tag{2.2}$$

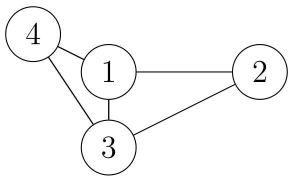
where $t > 0$ is another algorithmic *hyperparameter*. The choice of such a weight function is established by Belkin and Niyogi [20, 14] via connection of the Laplacian operator to heat flow. With edge matrix $E = \{e_{ij}\}$ and weight matrix $W = \{w_{ij}\}$, the weighted adjacency graph is then $G = (V, E, W)$.

If G is a connected graph, i.e., every node is reachable from every other node via edge connections, then the algorithm proceeds as follows (otherwise it proceeds on each connected component): compute the eigenvalues and eigenvectors for

$$Lf = \lambda D f, \tag{2.3}$$

where D is the diagonal matrix $D_{ii} = \sum_j W_{ij}$, and $L = D - W$ is the graph's Laplacian. The connection between the Laplacian operator on the sampled points X and the underlying manifold is well-founded by Belkin and Niyogi [21], and others [22, 23]. If $f_0, \cdots, f_{n-1}$ are solutions to equation 2.3, then the embedding f is taken as $\{f_1, \cdots, f_E\}$ such that for any $x_i \in X$, $f(x_i) = (f_{1i}, \cdots f_{Ei})$, where f_{ji} is the i^{th} entry in the j^{th} eigenvector.

As an example, we can consider a set of points lying in $\mathbb{R}^2$ that we wish to embed into $\mathbb{R}$, such as in Figure 2-1.



$$W = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \qquad D = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix} \qquad L = \begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 2 & -1 & 0 \\ -1 & -1 & 3 & -1 \\ -1 & 0 & -1 & 2 \end{bmatrix}$$

Figure 2-1. $X \subset \mathbb{R}^2$, connected with $k = 2$ and $t = \infty$.

Solving equation 2.3 for the example in Figure 2-1, we find the eigenvalues $\lambda_0 = 0$, $\lambda_1 = 1$, $\lambda_2 = \frac{4}{3}$, and $\lambda_3 = \frac{5}{3}$, with corresponding eigenvectors $f_0 = (1, 1, 1, 1)$, $f_1 = (0, -1, 0, 1)$, $f_2 = (-1, 0, 1, 0)$, and $f_3 = (-\frac{2}{3}, 1, -\frac{2}{3}, 1)$. The embedding is then $f = f_1 = (0, -1, 0, 1)$, and so $f(x_1) = 0$, $f(x_2) = -1$, $f(x_3) = 0$, and $f(x_4) = 1$.

The LE algorithm has two main computational tasks: finding the k-nearest neighbors during construction of the adjacency graph, and solving the eigenvalue problem when constructing the eigenmap embedding. Brute-force nearest-neighbor search takes $O(n^2)$ time as each sample is compared to every other sample when computing distances and neighbors, but more recent heuristic approaches, such as nearest neighbor descent [24], are able to achieve empirical run times of $O(n^{1.14})$. Similarly, solving the eigenvalue problem via brute-force takes $O(n^3)$ time, with efficient algorithms [25, 26] requiring only $O(n^2)$. In general, the eigenmap computation will be the rate-limiting step.

2.3 Uniform Manifold Approximation and Projection

The UMAP algorithm builds on LE in several important ways. Both algorithms begin in the same way: let $X = \{x_1, \cdots, x_n\}$ be a set of n points, and suppose that this data lies on a manifold within $\mathbb{R}^D$. UMAP makes three key assumptions as to the structure of this manifold:

- the data is uniformly distributed (in distance) on the manifold,
- the Riemannian metric is locally constant,
- the manifold is locally connected.

McInnes et al. [3] show that a metric may be constructed such that the data is approximately uniformly distributed (in distance) with regard to that metric.

Lemma 2.3.1 ([3]). *Let (M, g) be a Riemannian manifold in $\mathbb{R}^D$ and take $p \in M$. If g is locally constant about p in an open neighborhood U, then in a ball $B_p \subseteq U$ centered at p, the geodesic distance on the manifold from p to any point $q \in B_p$ is $\frac{1}{r} d_{\mathbb{R}^D}(p, q)$, where r is the radius of B_p in $\mathbb{R}^D$ and $d_{\mathbb{R}^D}$ is the existing metric on $\mathbb{R}^D$.*

This means that (except on the manifold boundary), any ball of fixed volume will contain approximately the same number of points. A ball centered at x_i that contains precisely x_i's $k-$nearest neighbors will (approximately) have fixed volume, and so by Lemma 2.3.1, the geodesic distance from x_i to its $k-$nearest neighbors is effectively constant. McInnes et al. [3] further show that this construction (in conjunction with the local connectivity assumption) corresponds with the Čech complex and captures the relevant topological structure.

UMAP uses this connection to update the construction of the adjacency graph as compared to LE by updating the weight function. For each $x_i \in X$, denote the set of $k-$nearest neighbors to x_i as $N_k(x_i)$, and define ρ_i and σ_i such that

$$\rho_i = \min\{d(x_i, x_{i_j}) | x_{i_j} \in N_k(x_i), d(x_i, x_{i_j}) > 0\}, \tag{2.4}$$

$$\sum_{j=1}^{k} \exp\left(\frac{-\max(0, d(x_i, x_{i_j}) - \rho_i)}{\sigma_i}\right) = \log_2(k). \tag{2.5}$$

Here ρ_i is the distance to the closest neighbor of x_i and σ_i is a normalization factor, which taken together enforce the local connectivity assumption. The weight function is then defined as

$$w_{ij} = \begin{cases} \exp\left(\frac{-\max(0,d(x_i,x_{i_j})-\rho_i)}{\sigma_i}\right), & x_j \in N_k(x_i), \\ 0, & \text{otherwise.} \end{cases} \tag{2.6}$$

Intuitively, the weight of an edge is the probability that it exists. If A is the weighted adjacency matrix, a weight $w_{ij} \in A$ represents the probability that an edge exists from x_i to x_j. UMAP constructs the symmetric matrix

$$B = A + A^T - A \circ A^T,$$

where "$\circ$" is the Hadamard (element-wise) product. The same weight $w_{ij} \in B$ represents the probability that an edge exists from both x_i to x_j and from x_j to x_i. The UMAP graph is then $G = (X, B)$. This weighted graph G encodes the topology of the source data, X, from which we then seek to find a set of data, $Y = \{y_1, \cdots, y_n\}$, such that the weighted graph H encoding the topology of Y is "similar" to G.

Interpreting the weights of G and H as the probability of an edge existing, these are Bernoulli variables as the edge either exists or it does not. Considering the set of all possible edge weights E, with weight functions $w_G(e)$ and $w_H(e)$ to represent the weight of such an edge in G and H respectively, UMAP then minimizes the Cross-Entropy between G and H as

$$C(G,H) = \sum_{e\in E}\left(w_G(e)\log\left(\frac{w_G(e)}{w_H(e)}\right) + (1-w_G(e))\log\left(\frac{1-w_G(e)}{1-w_H(e)}\right)\right) \tag{2.7}$$

$$= \sum_{e\in E}\Big(w_G(e)\log(w_G(e)) + (1-w_G(e))\log(1-w_G(e))\Big) \tag{2.8}$$

$$- \sum_{e\in E}\Big(w_G(e)\log(w_H(e)) + (1-w_G(e))\log(1-w_H(e))\Big). \tag{2.9}$$

Note that in 2.8, the weight function of G is fixed as it is constructed from the source data X, and so 2.7 minimizes precisely when 2.9 is minimized, that is

$$- \sum_{e\in E}\Big(w_G(e)\log(w_H(e)) + (1-w_G(e))\log(1-w_H(e))\Big). \tag{2.10}$$

UMAP employs stochastic gradient descent (SGD) to find such a minimizer. To compute the gradient required for SGD, UMAP makes use of a smooth approximation of $w_H(e_{ij})$ of the form

$$\Phi(e_{ij}) = \left(1 + a(\|x_i - x_j\|_2^{2b}\right)^{-1}, \tag{2.11}$$

where a, b are found by fitting 2.11 to 2.6. As an initial "guess" at Y, UMAP uses the lower-dimensional eigenmaps that LE computes via 2.3, and then minimizes according to 2.7. The resultant graph vertices are the optimal embedding. Figure 2-2 shows a simple comparison of LE versus UMAP embeddings, along with PCA.

2.4 Simultaneous Perturbation Stochastic Approximation

In the previous sections, we present brief summaries of the LE and UMAP algorithms for dimension reduction. At its core, UMAP added a critical step on top of the LE algorithm, namely the optimization of the low-dimensional representation. In this section we will introduce Simultaneous Perturbation Stochastic Approximation (SPSA) [4], a stochastic optimization algorithm. SPSA seeks to find the minimizer θ^* of a loss function $J(\theta)$

$$\theta^* = \arg\min_{\theta} J(\theta). \tag{2.12}$$

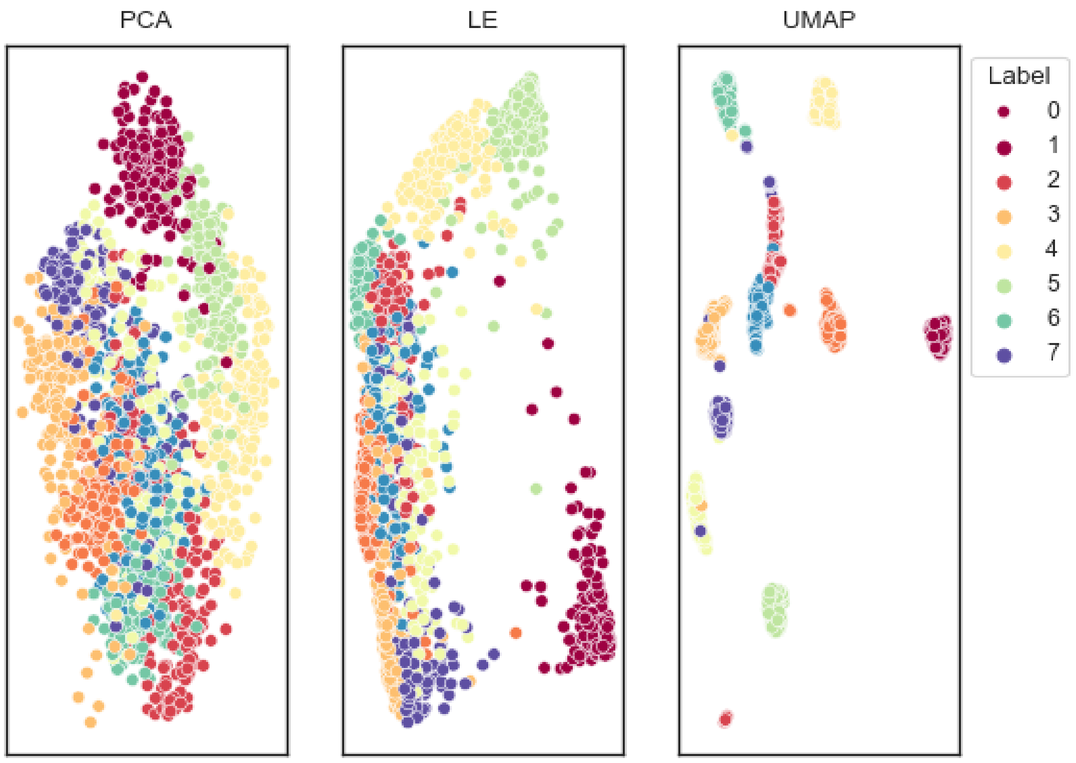


Figure 2-2. Comparison of PCA, LE, and UMAP embeddings of the Pendigits dataset. The LE embedding is able to generate some separation between classes, but UMAP does a much better job of bringing same-class points together, while separating the different classes.

From an initial guess $\hat{\theta}_0$, the algorithm progressively updates the current estimate $\hat{\theta}_k$ via estimates of the gradient of J in the general stochastic approximation form [27]

$$\hat{\theta}_{k+1} = \hat{\theta} - a_k \hat{g}_k(\hat{\theta}_k), \tag{2.13}$$

where $\hat{g}_k$ is the current gradient estimate and a_k is a nonnegative gain sequence. SPSA computes gradient estimates by perturbing the given parameter estimate $\hat{\theta}_k$ and evaluating the loss function at these perturbations. That is, if Δ_k is a zero-mean perturbation vector of the same dimension as $\hat{\theta}_k$, then

$$\hat{g}_k(\hat{\theta}_k) = \frac{J(\hat{\theta}_k + c_k\Delta_k) - J(\hat{\theta}_k - c_k\Delta_k)}{2c_k\Delta_k}, \tag{2.14}$$

where c_k is a nonnegative step sequence defined in [4]. Spall [4] defines specific conditions that these gain and step sequences must adhere to, namely:

- $a_k, c_k \to 0$
- $\sum a_k = \infty$
- $\sum \frac{a_k^2}{c_k^2} < \infty$

While many functional forms for a_k and c_k exist which satisfy these requirements, Spall recommends using

$$a_k = \frac{a}{(k+1+A)^{\alpha}} \qquad\qquad c_k = \frac{c}{(k+1)^{\gamma}}, \tag{2.15}$$

with a, A, α, c, and γ being optimization hyperparameters.

The gradient estimator constructed in 2.14 is shown to be an "almost unbiased" estimator for each m^{th} entry via a first-order Taylor expansion:

$$\begin{aligned}
E\Big[\hat{g}_{km}(\hat{\theta}_k)\Big|\hat{\theta}_k\Big] &= E\Big[\frac{J(\hat{\theta}_k + c_k\Delta_k) - J(\hat{\theta}_k - c_k\Delta_k)}{2c_k\Delta_{km}}\Big|\hat{\theta}_k\Big] \\
&\approx E\Big[\frac{J(\hat{\theta}_k) + c_k g(\hat{\theta}_k)^T\Delta_k) - [J(\hat{\theta}_k) - c_k g(\hat{\theta}_k)^T\Delta_k)]}{2c_k\Delta_{km}}\Big|\hat{\theta}_k\Big] \\
&= E\Big[\frac{2c_k\sum_i g_i(\hat{\theta}_k)\Delta_{ki}}{2c_k\Delta_{km}}\Big|\hat{\theta}_k\Big] \\
&= g_m(\hat{\theta}_k) + \sum_{i\neq m} g_i(\hat{\theta}_k)E[\frac{\Delta_{ki}}{\Delta_{km}}]
\end{aligned}$$

Assuming Δ_{ki} is a zero-mean random vector, is independent of Δ_{km}, and $E[\frac{1}{\Delta_{km}}]$ is finite, then $E[\frac{\Delta_{ki}}{\Delta_{km}}] = 0$. Thus

$$E\left[\hat{g}_{km}(\hat{\theta}_k)\middle|\hat{\theta}_k\right] \approx g_m(\hat{\theta}_k)$$

Spall [4] further notes that when compared to similar algorithms, such as Finite Difference Stochastic Approximation which require a number of loss function evaluations commensurate with the number of parameters, SPSA only requires two loss function measurements per gradient estimate.

Chapter 3

UMAP Optimization via SPSA

A problem with the UMAP SGD optimization is that while the weight function for a given graph H is easily understood (the probability that an edge exists between two points), its derivative ∇w_H is highly intractable with respect to the underlying edge. To circumvent this, UMAP constructs a smooth-approximation to perform Stochastic Gradient Descent for computing the minimizer (2.11) for 2.9. While this smooth approximation is differentiable, the derivative of even a good approximation may differ from the true gradient, resulting in poor performance. It is then desirable to avoid this construction.

Instead of constructing this smooth approximation, we may instead leverage the SPSA algorithm, which only requires two loss function measurements to compute a gradient estimate. Let

$$J(H) = -\sum_{e \in E} \Big(w_G(e) \log(w_H(e)) + (1 - w_G(e)) \log(1 - w_H(e)) \Big) \tag{3.1}$$

be the UMAP objective function; then we seek

$$H^* = \arg\min J(H). \tag{3.2}$$

With a_n, c_n as the usual SPSA gain sequences, then (3.2) is found by iterating

$$\begin{aligned} w_{H,n+1} &= w_{H,n} - a_n \hat{g}_n(w_{H,n}) \\ \hat{g}_n(w_{H,n})_i &= \frac{J(w_{H,n} + c_n \Delta_n) - J(w_{H,n} - c_n \Delta_n)}{2c_n(\Delta_n)_i}, \end{aligned}$$

where Δ_n is the random perturbation vector.

Figure 3-1 illustrates an example of the SPSA perturbation, as applied to the low-dimension UMAP embedding graph. For illustration, we consider a set of four points being embedded into $\mathbb{R}^2$. At this specific iteration, we highlight the perturbation of the second sample (2), which is perturbed in a positive, and negative direction. Note that all nodes are perturbed at once (simultaneously), but we only demonstrate a single node for convenience. This perturbation creates two graphs, for which loss measurements may be taken, and the gradient estimated per 2.14. Figure 3-2 shows a simple comparison of LE, UMAP, and UMAP-SPSA embeddings.

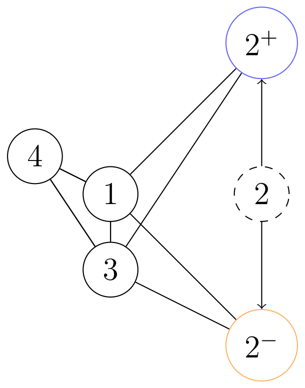


Figure 3-1. SPSA perturbation of UMAP graph

Figure 3-2 shows a simple comparison of LE, UMAP, and UMAP-SPSA embeddings.

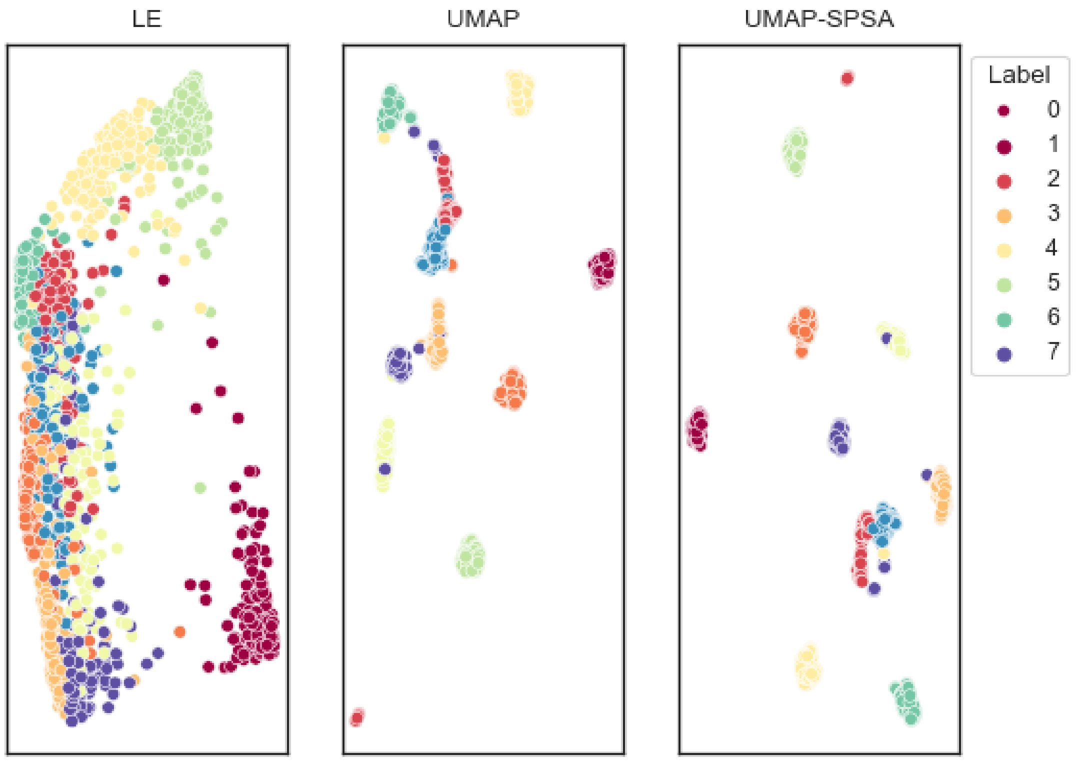


Figure 3-2. Comparison of LE, UMAP, and UMAP-SPSA embeddings of the Pendigits dataset. The LE embedding is able to generate some separation between classes, but UMAP does a much better job of bringing same-class points together, while separating the different classes. UMAP-SPSA generates comparable clustering and separation of samples as UMAP.

Chapter 4

Experiments and Results

The primary datasets used in this work are: Wisconsin Breast Cancer [28], Pendigits [29], MNIST [30], and FMNIST [31]. A summary of these datasets is provided in table 4-I.

Name	Samples	Dimensionality	Classes
Breast Cancer	569	30	2
Pendigits	1797	$64(8 \times 8)$	10
MNIST	70000	$784(28 \times 28)$	10
FMNIST	70000	$784(28 \times 28)$	10

Table 4-I. Summary of datasets

4.1 Computational Performance Comparison

Benchmarks were performed on a Windows 10 machine with a 3.6 GHz AMD Ryzen 5 3600 and 16 GB of DDR4 RAM. Scikit-learn [32] implementations of several dimension reduction techniques were compared against UMAP [3] and UMAP-SPSA, including PCA [11], TSNE [15], Local Linear Embedding [33], Laplacian Eigenmaps (Spectral Embedding) [14], Isomap [34], and Multidimensional Scaling [35]. Computational run-times were captured for these algorithms on the MNIST [30] dataset with a varing number of samples. A summary of runtime performance can be found in Table 4-II for the UMAP, UMAP-SPSA, and LE algorithms on several datasets.

For small subsample ($n < 1600$), all algorithms previously described are used to embed a subset of MNIST to 2D, with Figure 4-1 presenting the runtime scaling as number of subsamples is increased. All algorithms, except MDS, provide reasonable scaling at low sample sizes, but as sample size increases (5000 < n < 25000), TSNE and Local Linear Embedding scale poorly, as shown in Figure 4-2. Figure 4-3 shows the results for larger (n > 10000) numbers of samples for the UMAP, UMAP-SPSA, and PCA algorithms. While UMAP-SPSA is outperformed, in terms of recorded runtime, by UMAP, it is still faster than most other considered dimension reduction algorithms.

(Number of Samples, Dimension of Samples)	UMAP	UMAP-SPSA	Laplacian Eigenmaps
Breast Cancer (569, 30)	2.545734s ± 0.105s	1.985s ± 0.076s	0.026s ± 0.002s
Pendigits (1797, 64)	4.733s ± 0.375s	5.710s ± 0.114s	0.783s ± 0.020s
MNIST (70000, 784)	37.269s ± 0.567s	65.116s ± 0.909s	N/A
FMNIST (70000, 784)	31.420s ± 0.574s	69.261s ± 0.715s	N/A

Table 4-II. Runtime of UMAP, UMAP-SPSA, and LE on various datasets. LE runtimes are not computed for MNIST and FMNIST due to time required for convergence; UMAP authors reported times in excess of an hour, compared to seconds required by UMAP / UMAP-SPSA.

4.2 Quantitative Performance Comparison

We compare the PCA, UMAP, and UMAP-SPSA embeddings via a k-nearest neighbor classifier trained on the embedded data according to the procedure used by the UMAP authors with slight modification [3]. We embed the dataset using the various dimensionality reduction algorithms to $\mathbb{R}^d$ for several values of d, then train a kNN classifier for several values of k. The kNN classifier accuracy indicates how well the embedding has captured the local structure at small k values, while also indicating how well the embedding captured the

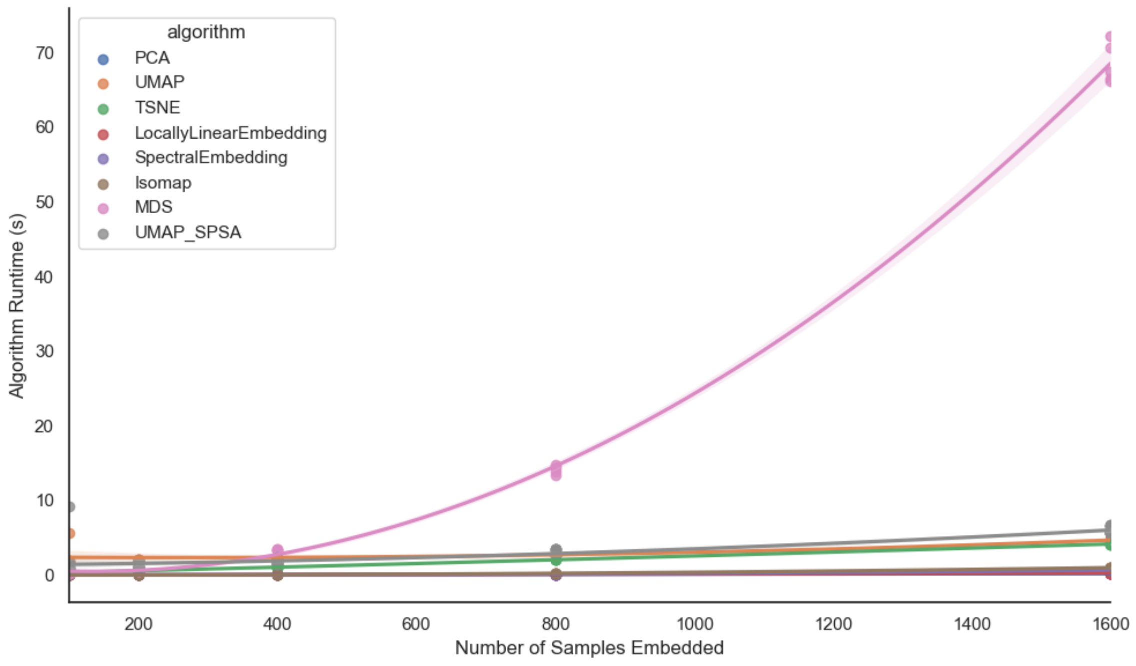


Figure 4-1. Scaling of embedding algorithm runtime with sample size

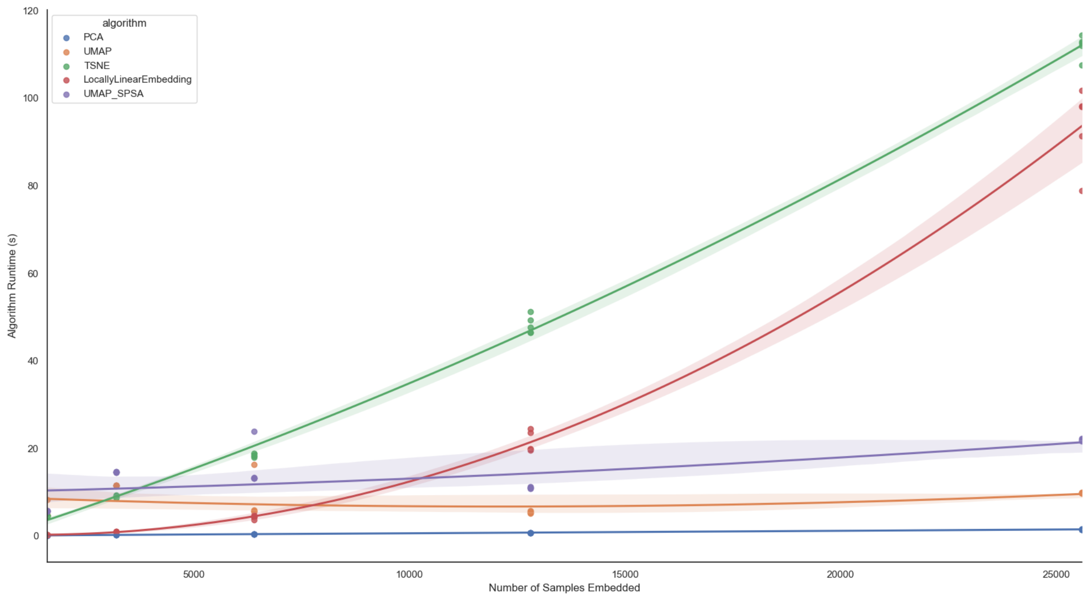


Figure 4-2. Scaling of fast embedding algorithm runtime with sample size

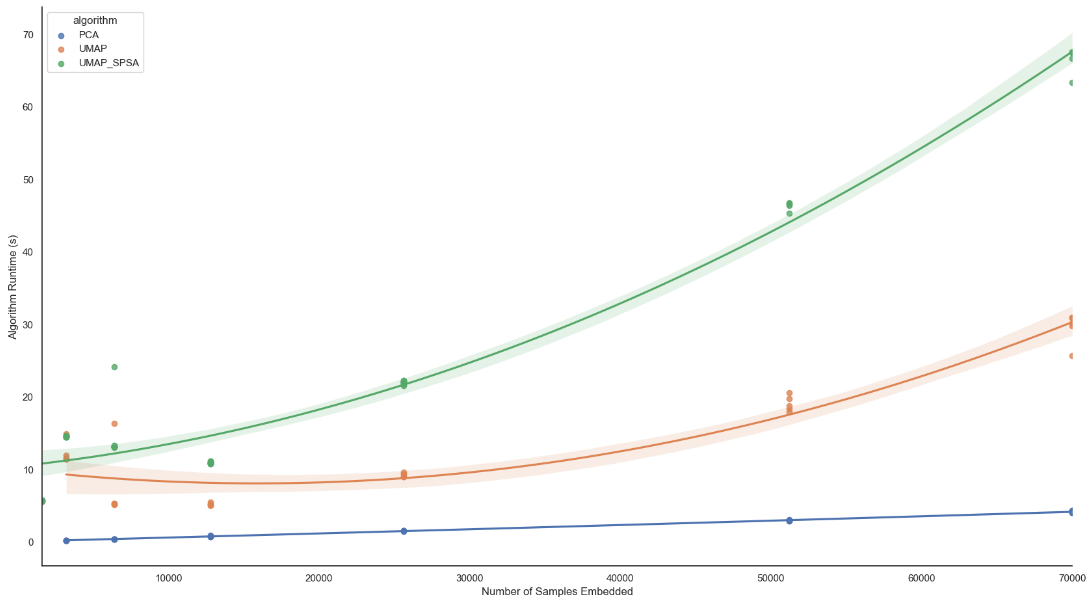


Figure 4-3. Scaling of very fast embedding algorithm runtime with sample size

global structure at large k values. Following the UMAP authors' protocol, we divide the datasets into small (Wisconsin Breast Cancer and Pendigits) for which smaller values of k are more reasonable and large (MNIST and FMNIST) for which larger values of k are more reasonable. A 10-fold cross-validation is used to compute 10 accuracy scores following each embedding. Across these datasets in general, UMAP-SPSA performs as well, or better than UMAP, or other comparable algorithms.

Figure 4-4 provides a facet grid of the accuracy results for the Cancer dataset. Each row corresponds to an embedding algorithm (PCA, UMAP, UMAP-SPSA), each column to the "k" value used in the kNN classifier, and each entry showing a plot of embedding dimension against classifier accuracy. Figure 4-5 summarizes the results for the Pendigits dataset, figure 4-6 summarizes the MNIST dataset, and figure 4-7 summarizes the FMNIST dataset. Overall, embedding dimension only affects the PCA accuracy, with classifier accuracy increasing as embedding dimension is increased, before leveling off. Broadly, the embedding accuracy between UMAP and UMAP-SPSA is comparable, with UMAP-SPSA exhibiting slightly superior performance on the Pendigits and FMNIST datasets.

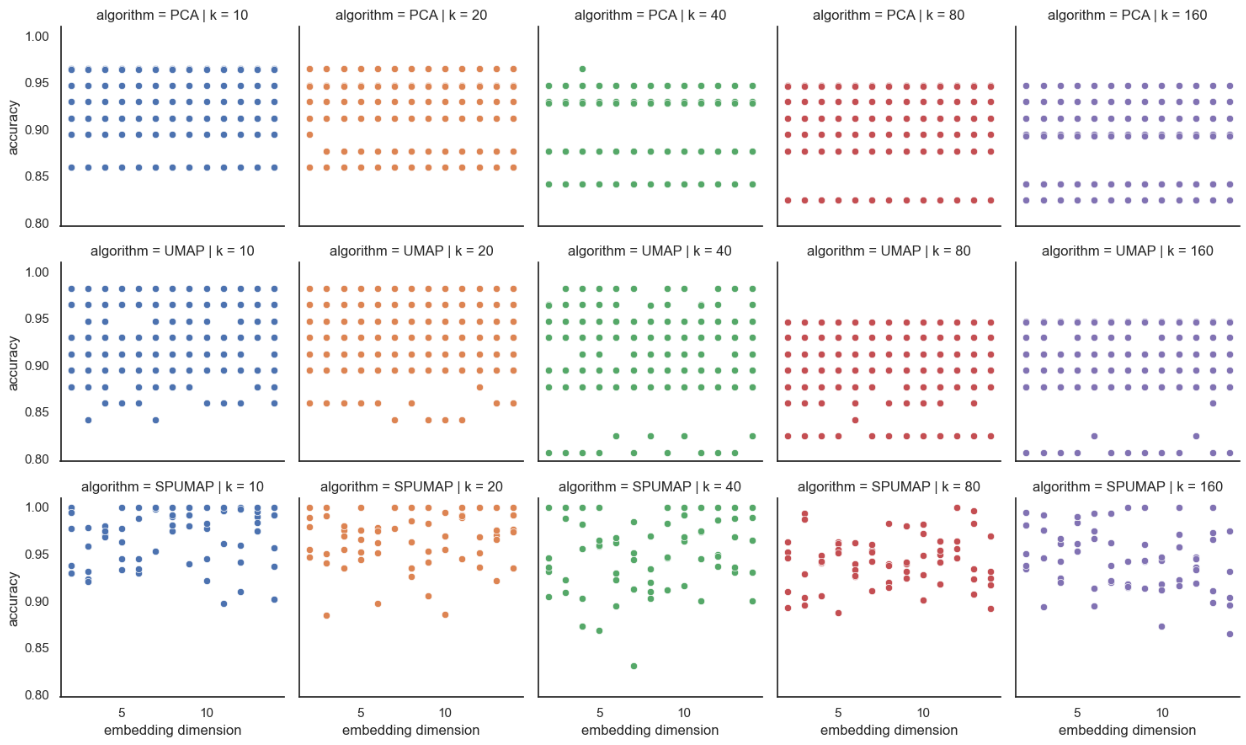


Figure 4-4. Wisconsin Breast Cancer Embedding Accuracy

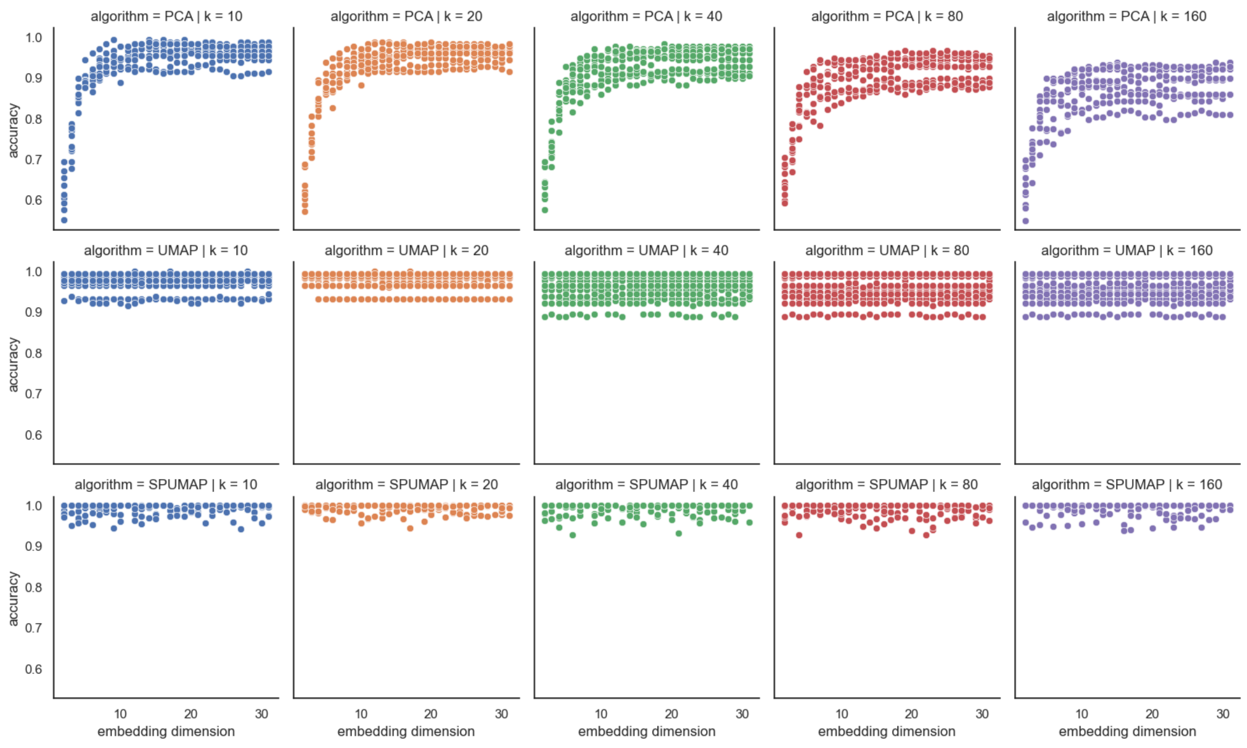


Figure 4-5. Pendigits Embedding Accuracy

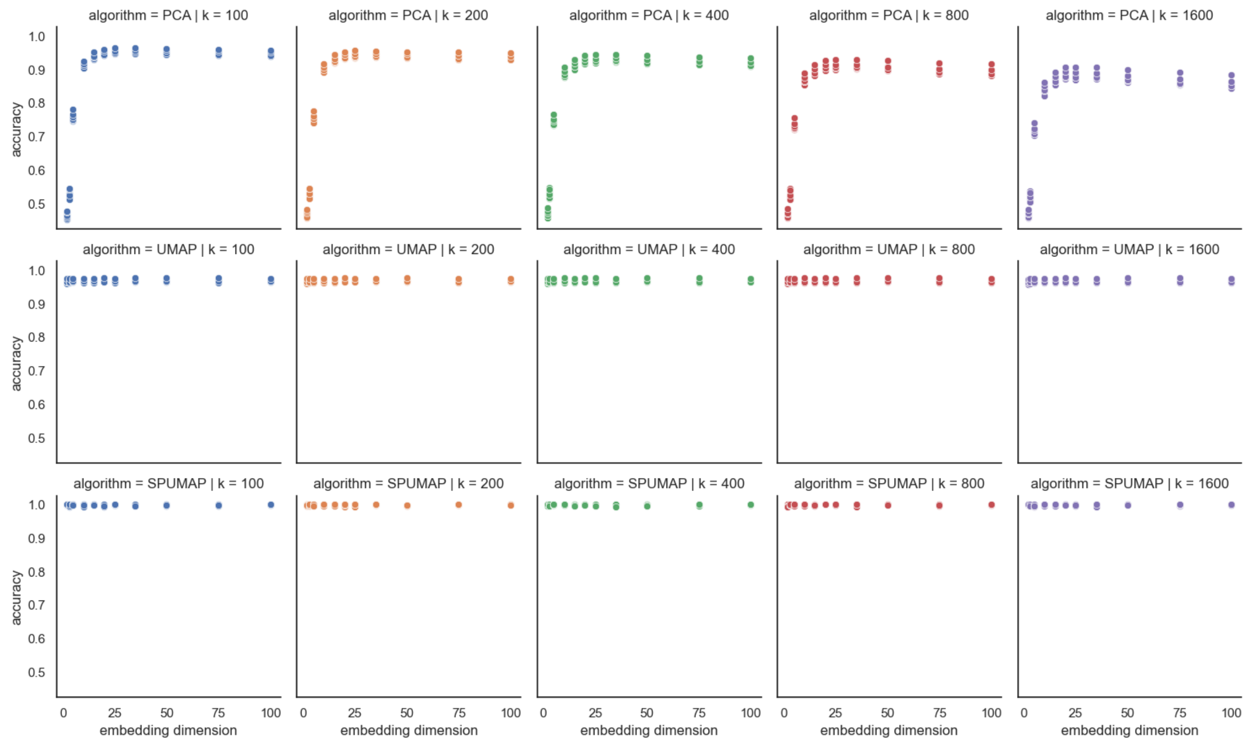


Figure 4-6. MNIST Embedding Accuracy

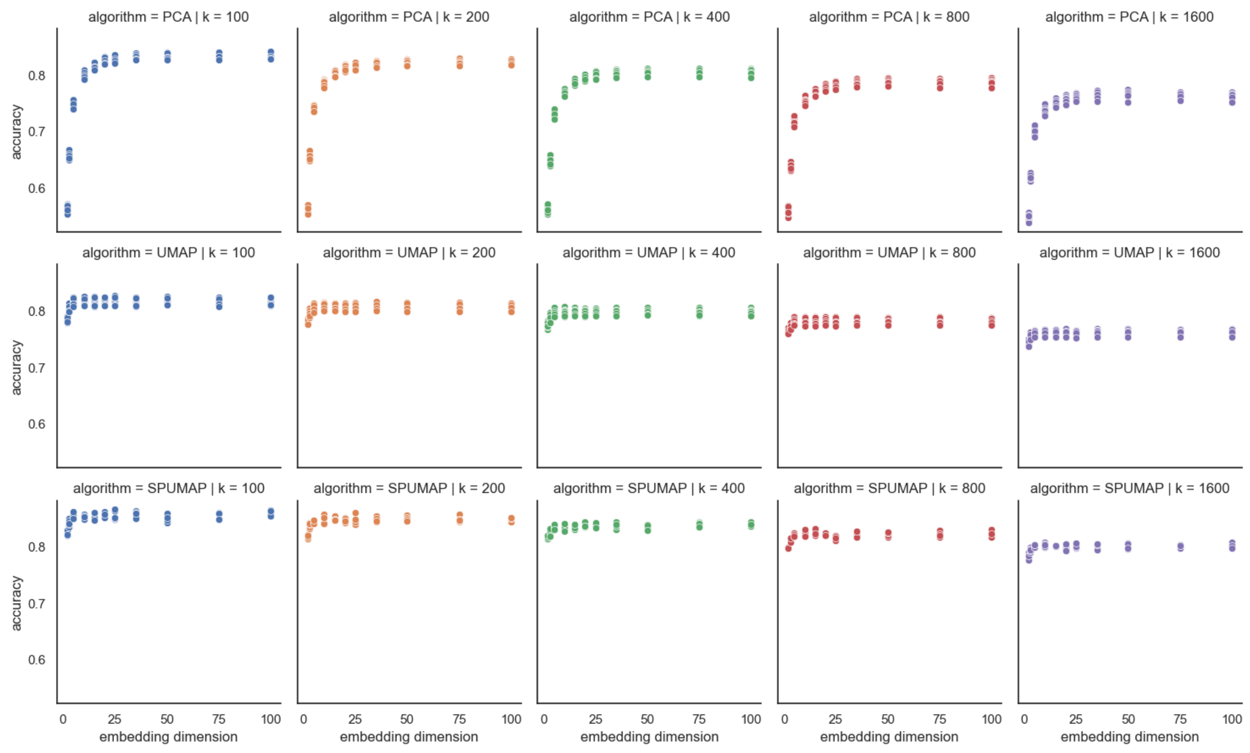


Figure 4-7. FMNIST Embedding Accuracy

Chapter 5

Conclusion and Future Work

5.1 Conclusion

This paper highlights the potential usefulness of the SPSA algorithm in optimizing the UMAP objective function. We can see that there is a trade-off in the algorithms computational run-time, and embedding accuracy. Through the work of Spall, we show that a gradient estimate of the UMAP graph's cross-entropy may be constructed using only two loss measurements at each iteration, regardless of the number, or dimension, of the samples to be embedded.

Our main contribution in this paper was the UMAP-SPSA algorithm which we introduced in Chapter 3. We were able to show that this approach, while slower than UMAP, is still computationally competitive against other dimension reduction techniques, while maintaining embedding accuracy.

5.2 Future Work

We believe there are still research paths to explore regarding the UMAP-SPSA algorithm. While we introduce the algorithm and demonstrate its relative performance as compared to UMAP and other dimension reduction techniques, the question remains as to the relationship of the bias-variance trade-off between the original UMAP gradient estimate and the SPSA gradient estimate. It may be possible to express the bias and variance of the different gradient estimates using common terms, and establish a bound relating the two. Spall [4] established the expression for the bias of the SPSA gradient estimate in terms of the SPSA step sequence, perturbation vector, perturbed parameters, and the loss function.

A one-measurement version of SPSA [36] also exists which leverages a single loss function measurement at each iteration to construct the gradient estimate. While the one-measurement SPSA algorithm is asymptotically superior to the two-measurement SPSA for a specific class of problems, it is not currently known whether the UMAP optimization falls within this class. However, we hypothesize that using the one-measurement form of SPSA would bring the UMAP-SPSA algorithm's runtime in line with that of UMAP's. Additionally, a second-order SPSA method [37] exists which leverages multiple loss measurements at each iteration to estimate both the gradient and hessian of the loss function. The second-order method accelerates convergence to the minimizer as compared to the first-order two-measurement SPSA method) at the cost of requiring more loss measurements at each iteration.

Overall, our work has shown that leveraging first-order SPSA to optimize the UMAP objective function is beneficial. From these results, we hypothesize it is worthwhile to pursue further SPSA based modifications to UMAP.

Appendix A

Topological Background

A.1 Topological Spaces

Let T be a set. A collection τ of subsets of T is a **topology** on T, if

- both T and $\emptyset$, the empty set, belong to τ,
- the arbitrary union of sets in τ belongs to τ,
- the finite intersection of sets in τ belongs to τ.

A **topological space** (T, τ) is a set T paired with a topology τ on T. For a set T, multiple topologies may exists, including the **trivial topology** containing only the empty set and T itself, and the **discrete topology** containing every subset of T. A subset of T belonging to τ is said to be **open**. For any open subset $U \in \tau$, the complement of U is said to be **closed**.

Within a topological space (T, τ), one can consider a sequence of points $\{x_n\}$, with $x_n \in T$. The **N-tail of** x_n is the collection of all points $\{x_i : i > N\}$. A sequence of points $\{x_n\}$ in T is said to **converge to** $x \in T$ if for every open set $U \in \tau$ containing x, there is an integer N such that $x_i \in U$ for all $i > N$. In other words, given any open set U containing x, there exists an $N \in \mathbb{N}$ such that the N-tail of x_n is contained in U. One such example is the real-line $\mathbb{R}$, paired with the topology of arbitrary union of open intervals, and the sequence $\{\frac{1}{n}\}$. This sequence converges to 0, since for every open interval containing 0, i.e., (a, b) with $a < 0 < b$ and $a, b \in \mathbb{R}$, there will exist N such that $a < 1/n < b$ for $n > N$.

Now consider two topological spaces (T_1, τ_1) and (T_2, τ_2). A **function** f from T_1 to T_2, $f : T_1 \to T_2$, is a mapping that assigns to each element in T_1 a unique element in T_2. The simplest function is the identify function, $id_T : T \to T$, which maps each element of a space T to the same element in T, e.g. $id_X(x) = x$. A function $f : T_1 \to T_2$ is said to be **injective** if for all $x, y \in T_1$, $f(x) = f(y)$ implies $x = y$. A function $f : T_1 \to T_2$ is said to be **surjective** if for all $y \in T_2$, there exists $x \in T_1$ such that $f(x) = y$. A function that is injective and surjective is said to be **bijective**. If $f : T_1 \to T_2$ and $g : T_2 \to T_1$, then g is said to be the **inverse** of f if $f \circ g = id_{T_2}$ and $g \circ f = id_{T_1}$, where $\circ$ denotes function composition, i.e., $(g \circ f)(x) = g(f(x))$. Similarly, f is the inverse of g.

Lemma A.1.1. *A function $f : T_1 \to T_2$ has an inverse if and only if it is bijective.*

Proof. Let $f : T_1 \to T_2$ be bijective. As f is surjective, for all $y \in T_2$, there exists $x \in T_1$ such that $f(x) = y$, and as f is injective, this x is unique. Define $g : T_2 \to T_1$ such that $g(y) = x$ for each such x, y pairing. This g is well-defined, and $(f \circ g)(y) = id_{T_2}$ and $(g \circ f) = id_{T_1}$.

Now let f have an inverse g. Choose $y \in T_2$ and let $x \in T_1$ be such that $g(y) = x$. Applying f to both sides yields $y = f(x)$, as $(f \circ g) = id_{T_2}$, and so f is surjective. Take $x_1, x_2 \in T_1$ such that $f(x_1) = f(x_2)$. Let $y \in T_2$ be such that $f(x_1) = f(x_2) = y$. As g is the inverse of f, $g(y) = (g \circ f)(x_1) = (g \circ f)(x_2) \to x_1 = x_2$, and so f is injective. $\square$

A function $f : T_1 \to T_2$ is said to be **continuous** if for every open set $U \subseteq T_2$, $f^{-1}(U) = \{x \in T_1 : f(x) \in U\}$ is open in T_1. A continuous bijective function $f : T_1 \to T_2$, with continuous inverse $f^{-1} : T_2 \to T1$, is a **homeomorphism**. If such a function exists between two topological spaces T_1 and T_2, then T_1 is **homeomorphic** to T_2, and vice versa. In other words, if $f : T_1 \to T_2$ is a homeomorphism, we may continuously deform T_1 into T_2 via f, and T_2 into T_1 via f^{-1}.

Given two continuous functions, $f, g : T_1 \to T_2$, one can consider the deformation of f into g. That is, consider the function $h : T_1 \times [0, 1] \to T_2$ such that $h(x, 0) = f(x)$ and $h(x, 1) = g(x)$. Then $h(x, t)$ deforms $f(x)$ into $g(x)$ as t goes from zero to one. If h is continuous, then h is said to be a **homotopy**. If there exist continuous functions $f : T_1 \to T_2$ and $g : T_2 \to T_1$ such that there exists a homotopy between $f \circ g$ and id_{T_1} and a homotopy between $g \circ f$ and id_{T_2}, then T_1 and T_2 are said to be **homotopy-equivalent**. Intuitively, we may deform T_1 into T_2 by stretching or shrinking T_1 into T_2.

Lemma A.1.2. *If topological spaces T_1 and T_2 are homeomorphic, then they are also homotopy-equivalent.*

Proof. Let T_1, T_2 be two homeomorphic topological spaces, with homeomorphism $f : T_1 \to T_2$. As f is bijective, $(f^{-1} \circ f)(x) = x$ for all $x \in T_1$, and so $f^{-1} \circ f = id_{T_1}$. Let $h(x,t) = (f^{-1} \circ f)(x)$, then $h(x,0) = (f^{-1} \circ f)(x)$ and $h(x,1) = (f^{-1} \circ f)(x) = id_{T_1}(x)$, and so h is a homotopy between $f^{-1} \circ f$ and id_{T_1}. The same may be shown for $f \circ f^{-1}$ and id_{T_2}. Thus T_1 and T_2 are homotopy equivalent. □

A.2 Topological Bases

When considering a space T, one may have a preconceived notion as to what sets should be open in that space, e.g., the canonical open intervals in $\mathbb{R}$. From this collection, one seeks to *generate* a topology guaranteeing the openness of said collection. For a collection $\mathcal{B}$ of open sets in a topological space (T, τ), $\mathcal{B}$ is said to be a **base** of τ if every set in τ may be expressed as the union of sets from $\mathcal{B}$.

Lemma A.2.1. *Given a topological space (T, τ), a collection $\mathcal{B}$ of open sets is a base of τ if and only if for each $U \in \tau$ and each $x \in U$, there exists $B \in \mathcal{B}$ such that $x \in B \subseteq U$.*

Proof. Let (T, τ) be a topological space, and let $\mathcal{B}$ be a collection of open sets in T. If $\mathcal{B}$ is a base, then U is the union of sets in $\mathcal{B}$ (as $\mathcal{B}$ is a base). Let x belong to U, then one such $B \in \mathcal{B}$ must exist that contains x as part of such a union; denote this set as B^*. Then $x \in B^* \subseteq U$.

Now suppose that for each $U \in \tau$ and $x \in U$ there exists $B_x \in \mathcal{B}$ such that $x \in B_x \subseteq U$. Now, U is the union of the B_x, which completes the proof. □

A topological space (T, τ) is **second-countable** if there exists an at most countably infinite base of τ. By limiting the size of the base to being at most countably infinite, a second-countable space restricts how "large" (in an abstract sense) τ may be. We now present two important theorems regarding second-countable spaces.

Lemma A.2.2 (Lindelöf's Lemma)**.** *Let (T, τ) be a second-countable topological space, then every open cover of T contains an at most countable subcover.*

Proof. Let (T, τ) be a second-countable topological space, and let $\mathcal{G}$ be an open covering of T. As T is second-countable, there exists an at most countable base $\mathcal{B}$ of τ. As every $G \in \mathcal{G}$ is open, and G is the union of sets in $\mathcal{B}$, $\mathcal{G}$ may be rewritten as the union of sets in $\mathcal{B}$. But $\mathcal{B}$ is at most countable, and thus $\mathcal{G}$ may be rewritten as the union of an at most countable number of open sets. □

A set $D \subseteq T$ is said to be *dense in* T, if for every non-empty open set $U \subseteq T$, there exists a point in D that is also contained in U; i.e., D and U have non-empty intersection. If T has a dense set D that is at most countably infinite, then T is said to be *separable*. One such example of a separable space is $\mathbb{R}$ under the usual topology (of arbitrary union of open intervals). $\mathbb{Q}$ is countably infinite, and every open interval in $\mathbb{R}$ must contain a rational number (by the Archimedean property).

Lemma A.2.3. *Let (T, τ) be a second-countable topological space, then T is separable.*

Proof. Let (T, τ) be a second-countable topological space. As T is second-countable, there exists a countable base $\mathcal{B}$ of τ. For each $B_i \in \mathcal{B}$, choose one point $d_i \in B_i$, and denote the set of all d_i as D. Clearly D is countable, as one d_i is chosen for each set in $\mathcal{B}$, a countable set. For any non-empty open set U in T, and any $x \in U$, there exists $B \in \mathcal{B}$ such that $x \in B \subset U$

(by Lemma 1.1). Thus U must also contain an element from D, as it contains a set from $\mathcal{B}$ as a subset, and so D is dense in T. $\square$

A.3 Hausdorff Spaces

The previous subsection considered topological spaces under a restriction on how "big" (in an abstract sense) τ could be. When a space is restricted to being second-countable, two nice properties were shown to follow: the space is separable and Lindelöf. When considered under a different restriction, namely when τ cannot be too "small", different, but similarly nice properties exist.

Consider a topological space (T, τ). T is said to be a **Hausdorff** space, if for any two distinct points, $u, v \in T$, there exist open sets $U, V \in \tau$, such that $u \in U$ and $v \in V$, with U and V being disjoint. This implies uniqueness of limits. Otherwise, one could have a sequence x_n and two distinct points x and y, such that every open set of x contains elements of x_n, and every open set of y contains elements of x_n; in this sense, x_n converges to "both" x and y.

A.4 Topological Manifolds

Consider a second-countable Hausdorff space M. A **chart** (U, ϕ) is an open subset U of M, with a homeomorphism ϕ from U onto an open subset of $\mathbb{R}^n$. (U, ϕ) is said to be a **chart at p**, for $p \in M$ if $p \in U$. An **atlas** is a collection of charts, $\{(U_i, \phi_i)\}_{i \in I}$, such that the collection $\{U_i\}_{i \in I}$ covers M.

Given an atlas $\mathcal{A}$, and two charts, $(U_i, \phi_i), (U_j, \phi_j) \in \mathcal{A}$, if $U_i \cap U_j \neq \emptyset$, then

$$\phi_i^j : \phi_i(U_i \cap U_j) \to \phi_j(U_i \cap U_j) \qquad \phi_j^i : \phi_j(U_i \cap U_j) \to \phi_i(U_i \cap U_j)$$

are **transition maps**. Note that $\phi_i^j = \phi_j \circ \phi_i^{-1}$ and $\phi_j^i = \phi_i \circ \phi_j^{-1}$, and thus $\phi_i^j = (\phi_j^i)^{-1}$. Intuitively, *charts* describe the mapping of regions of M to $\mathbb{R}^n$, while *transition maps* describe the transition between regions of M, precisely on the overlap of the regions.

M is said to be a **topological n-manifold** if an atlas of M exists. A topological manifold with an atlas whose transition maps are all differentiable is a **differentiable n-manifold**; a topological manifold with an atlas whose transition maps are all smooth (continuously differentiable) is a **smooth n-manifold**. Intuitively, this means that each point in M has a local neighborhood that can be continuously (or smoothly) deformed into $\mathbb{R}^n$.

A.5 Tangent Spaces and Riemannian Manifolds

Just as a tangent may be constructed at a point along a curve in $\mathbb{R}^n$, one may consider the tangents of a point on a manifold. Consider a smooth n-manifold M, and choose a point $p \in M$, with chart (U_p, ϕ_p) at p. Now consider a curve lying on M which passes through p, $\gamma : (-1,1) \subset \mathbb{R} \to M$ with $\gamma(0) = p$, such that $\phi_p \circ \gamma : (-1,1) \subset \mathbb{R} \to \mathbb{R}^n$ is differentiable in the usual sense. The **tangent vector of M at p along γ** is the derivative of $\phi_p \circ \gamma$ at 0. The set of all tangent vectors of M at p, denoted T_pM, is the **tangent space of M at p**. Intuitively, the tangent space of a point $p \in M$ is constructed by taking all curves on M which pass through p, and taking the tangent of those curves at p.

Consider now a real, smooth manifold M. For each point $p \in M$, equip the tangent space T_pM with an inner-product $g_p : T_pM \times T_pM \to \mathbb{R}$ such that:

- $g_p(u, v) = g_p(v, u)$, for all $u, v \in T_pM$,
- $g_p(u, u) \geq 0$, for all $u \in T_pM$,
- $g_p(u, u) = 0$ if and only if $u = 0$.

Then M is said to be a **Riemannian manifold**, and g_p is the **Riemannian metric at p**; the collection of all Riemannian metrics on M is known as the **Riemannian metric**.

Bibliography

[1] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. "Automatic Subspace Clustering of High Dimensional Data." In: *Data Mining and Knowledge Discovery* 11.1 (2005), 5–33. DOI: 10.1007/s10618-005-1396-1.

[2] Hans-Peter Kriegel, Peer Kröger, and Arthur Zimek. "Clustering high-dimensional data." In: *ACM Transactions on Knowledge Discovery from Data* 3.1 (2009), 1–58. DOI: 10.1145/1497577.1497578.

[3] Leland McInnes, John Healy, and James Melville. *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction*. 2020. arXiv: 1802.03426 [stat.ML].

[4] James C. Spall. *Introduction to stochastic search and optimization: estimation, simulation, and control*. Wiley-Interscience, 2003.

[5] Herbet Pang and Tiejun Tong. "Recent Advances in Discriminant Analysis for High-dimensional Data Classification." In: *Journal of Biometrics amp; Biostatistics* 03.02 (2012). DOI: 10.4172/2155-6180.1000e106.

[6] Janaina Mourão-Miranda, Arun L.W. Bokde, Christine Born, Harald Hampel, and Martin Stetter. "Classifying brain states and determining the discriminating activation patterns: Support Vector Machine on functional MRI data." In: *NeuroImage* 28.4 (2005). Special Section: Social Cognitive Neuroscience, pp. 980–995. ISSN: 1053-8119. DOI: https://doi.org/10.1016/j.neuroimage.2005.06.070. URL: https://www.sciencedirect.com/science/article/pii/S1053811905004787.

[7] Swarajya Lakshmi V. Papineni, Snigdha Yarlagadda, Harita Akkineni, and A. Mallikarjuna Reddy. *Big Data Analytics Applying the Fusion Approach of Multicriteria Decision Making with Deep Learning Algorithms*. 2021. arXiv: 2102.02637 [cs.LG].

[8] Liangyun Liu, Jihua Wang, Yansong Bao, Wenjiang Huang, Zhihong Ma, and Chunjiang Zhao. "Predicting winter wheat condition, grain yield and protein content using multi-temporal EnviSat-ASAR and Landsat TM satellite images." In: *International Journal of Remote Sensing* 27.4 (2006), pp. 737–753. DOI: 10.1080/01431160500296867. eprint: https://doi.org/10.1080/01431160500296867. URL: https://doi.org/10.1080/01431160500296867.

[9] Richard Bellman. *Dynamic Programming*. Princeton, NJ: Princeton University Press, 2010.

[10] Trevor Hastie, Jerome Friedman, and Robert Tisbshirani. *The Elements of statistical learning: data mining, inference, and prediction*. 2nd ed. Springer, 2017.

[11] Karl Pearson. "On lines and planes of closest fit to systems of points in space." In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2.11 (1901), pp. 559–572.

[12] Charles Spearman. "General intelligence objectively determined and measured." In: *American Journal of Psychology* 15.2 (1904), pp. 201–293.

[13] Ronald Fisher. "The Use of Multiple Measurements in Taxonomic Problems." In: *Annals of Eugenics* 7.7 (1936), pp. 179–188.

[14] Mikhail Belkin and Partha Niyogi. "Laplacian Eigenmaps for Dimensionality Reduction and Data Representation." In: *Neural Computation* 15.6 (2003), 1373–1396. DOI: 10.1162/089976603321780317.

[15] Laurens van der Maaten and Geoffrey Hinton. "Visualizing Data using t-SNE." In: *Journal of Machine Learning Research* 9 (2008), pp. 2579–2605. URL: http://www.jmlr.org/papers/v9/vandermaaten08a.html.

[16] Robert W. Ghrist. *Elementary applied topology.* 1st ed. United States: CreateSpace, 2014.

[17] Partha Niyogi, Stephen Smale, and Shmuel Weinberger. "Finding the Homology of Submanifolds with High Confidence from Random Samples." In: *Discrete amp; Computational Geometry* (2006). DOI: 10.1007/s00454-006-1250-7.

[18] J. R. Munkres. *Algebraic topology.* Addison-Wesley, 1984.

[19] John Gilbert Hocking and Gail S. Young. *Topology.* Dover Publications, 1988.

[20] Isaac Chavel. *Eigenvalues in Riemannian geometry.* Academic Press, 1984.

[21] Mikhail Belkin and Partha Niyogi. "Towards a Theoretical Foundation for Laplacian-Based Manifold Methods." In: *Learning Theory* (2005), 486–500. DOI: 10.1007/11503415_33.

[22] Matthias Hein, Jean-Yves Audibert, and Ulrike von Luxburg. "From Graphs to Manifolds – Weak and Strong Pointwise Consistency of Graph Laplacians." In: *Learning Theory* (2005), 470–485. DOI: 10.1007/11503415_32.

[23] A. Singer. "From graph to manifold Laplacian: The convergence rate." In: *Applied and Computational Harmonic Analysis* 21.1 (2006), 128–134. DOI: 10.1016/j.acha.2006.03.004.

[24] Wei Dong, Charikar Moses, and Kai Li. "Efficient k-nearest neighbor graph construction for generic similarity measures." In: *Proceedings of the 20th international conference on World wide web - WWW '11* (2011). DOI: 10.1145/1963405.1963487.

[25] V.N. Kublanovskaya. "On some algorithms for the solution of the complete eigenvalue problem." In: *USSR Computational Mathematics and Mathematical Physics* 1.3 (1962), 637–657. DOI: 10.1016/0041-5553(63)90168-x.

[26] James W. Demmel. *Applied numerical linear algebra.* Universities Press, 2018.

[27] Herbert Robbins and Sutton Monro. "A Stochastic Approximation Method." In: *The Annals of Mathematical Statistics* 22.3 (1951), pp. 400 –407. DOI: 10.1214/aoms/1177729586. URL: https://doi.org/10.1214/aoms/1177729586.

[28] Abien Fred M. Agarap. "On Breast Cancer Detection: An Application of Machine Learning Algorithms on the Wisconsin Diagnostic Dataset." In: *Proceedings of the 2nd International Conference on Machine Learning and Soft Computing.* ICMLSC '18. Phu Quoc Island, Viet Nam: ACM, 2018, pp. 5–9. ISBN: 978-1-4503-6336-5. DOI: 10.1145/3184066.3184080. URL: http://doi.acm.org/10.1145/3184066.3184080.

[29] Dheeru Dua and Casey Graff. *UCI Machine Learning Repository.* 2017.

[30] Yann LeCun and Corinna Cortes. "MNIST handwritten digit database." In: (2010). URL: http://yann.lecun.com/exdb/mnist/.

[31] Han Xiao, Kashif Rasul, and Roland Vollgraf. *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms.* cite arxiv:1708.07747Comment: Dataset is freely available at https://github.com/zalandoresearch/fashion-mnist Benchmark is available at http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/. 2017. URL: http://arxiv.org/abs/1708.07747.

[32] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. "Scikit-learn: Machine Learning in Python." In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[33] Sam T. Roweis and Lawrence K. Saul. "Nonlinear Dimensionality Reduction by Locally Linear Embedding." In: *Science* 290.5500 (2000), pp. 2323–2326. ISSN: 0036-8075. DOI: 10.1126/science.290.5500.2323. eprint: http://science.sciencemag.org/content/290/5500/2323.full.pdf. URL: http://science.sciencemag.org/content/290/5500/2323.

[34] Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. "A Global Geometric Framework for Nonlinear Dimensionality Reduction." In: *Science* 290 (2000), pp. 2319 –2323.

[35] J.B. Kruskal. "Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis." In: *Psychometrika* 29.1 (1964), pp. 1–27.

[36] James C. Spall. "A one-measurement form of simultaneous perturbation stochastic approximation." In: *Automatica* 33.1 (1997), pp. 109–112. ISSN: 0005-1098. DOI: https://doi.org/10.1016/S0005-1098(96)00149-5. URL: https://www.sciencedirect.com/science/article/pii/S0005109896001495.

[37] J.c. Spall. "Adaptive stochastic approximation by the simultaneous perturbation method." In: *Proceedings of the 37th IEEE Conference on Decision and Control (Cat. No.98CH36171)* (2000). DOI: 10.1109/cdc.1998.761833.