Exploring New Forms of Random Projections for Prediction and Dimensionality Reduction in Big-Data Regimes

by

Amir-Hossein Karimi

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Masters of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2018

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

The following two papers are used in this thesis. They are described below:

A. H. Karimi, M. J. Shafiee, A. Ghodsi, and A. Wong, "Synthesizing deep neural network architectures using biological synaptic strength distributions," in *Computational Cognitive Neuroscience (CCN)*, 2017.

This paper is incorporated in Chapter 3 of this thesis.

Contributor	Statement of Contribution
A. H. Karimi (Candidate)	Conceptual design (60%) Writing and editing (70%) Experimental design and analysis (100%)
M. J. Shafiee	Conceptual design (10%) Writing and editing (10%)
A. Ghodsi	Conceptual design (15%) Writing and editing (10%)
A. Wong	Conceptual design (15%) Writing and editing (10%)

A. H. Karimi, M. J. Shafiee, A. Ghodsi, and A. Wong, "Ensembles of Random Projections for Nonlinear Dimensionality Reduction," in *Journal of Computational Vision and Imaging Systems (JCVIS)*, vol. 3, number 1, 2017.

This paper is incorporated in Chapter 4 of this thesis.

Contributor	Statement of Contribution
A. H. Karimi (Candidate)	Conceptual design (60%) Writing and editing (70%) Experimental design and analysis (100%)
M. J. Shafiee	Conceptual design (10%) Writing and editing (10%)
A. Ghodsi	Conceptual design (15%) Writing and editing (10%)
A. Wong	Conceptual design (15%) Writing and editing (10%)

Abstract

The story of this work is dimensionality reduction. Dimensionality reduction is a method that takes as input a point-set $\mathcal{P}$ of n points in $\mathbb{R}^d$ where d is typically large and attempts to find a lower-dimensional representation of that dataset, in order to ease the burden of processing for down-stream algorithms. In today's landscape of machine learning, researchers and practitioners work with datasets that either have a very large number of samples, and or include high-dimensional samples. Therefore, dimensionality reduction is applied as a pre-processing technique primarily to overcome the *curse of dimensionality*.

Generally, dimensionality reduction improves time and storage space required for processing the point-set, removes multi-collinearity and redundancies in the dataset where different features may depend on one another, and may enable simple visualizations of the dataset in 2-D and 3-D making the relationships in the data easy for humans to comprehend. Dimensionality reduction methods come in many shapes and sizes. Methods such as Principal Component Analysis (PCA), Multi-dimensional Scaling, IsoMaps, and Locally Linear Embeddings are amongst the most commonly used method of this family of algorithms. However, the choice of dimensionality reduction method proves critical in many applications as there is no one-size-fits-all solution, and special care must be considered for different datasets and tasks. Furthermore, the aforementioned popular methods are data-dependent, and commonly rely on computing either the Kernel / Gram matrix or the covariance matrix of the dataset. These matrices scale with increasing number of samples and increasing number of data dimensions, respectively, and are consequently poor choices in today's landscape of big-data applications.

Therefore, it is pertinent to develop new dimensionality reduction methods that can be efficiently applied to large and high-dimensional datasets, by either reducing the dependency on the data, or side-stepping it altogether. Furthermore, such new dimensionality reduction methods should be able to perform on par with, or better than, traditional methods such as PCA. To achieve this goal, we turn to a simple and powerful method called random projections.

Random projections are a simple, efficient, and data-independent method for stably embedding a point-set $\mathcal{P}$ of n points in $\mathbb{R}^d$ to $\mathbb{R}^k$ where d is typically large and k is on the order of $\log n$. Random projections have a long history of use in dimensionality reduction literature with great success. In this work we are inspired to build on the ideas of random projection theory, and extend the framework and build a powerful new setup of random projections for large high-dimensional datasets, with comparable performance to state-of-the-art data-dependent and nonlinear methods. Furthermore, we study the use of random projections in domains other than dimensionality reduction, including prediction, and show

the competitive performance of such methods for processing small dataset regimes.

Acknowledgements

First and foremost, I would like to express my sincerest appreciation to my advisors, Professor Ali Ghodsi and Professor Alexander Wong. I have been amazingly fortunate to have advisors who gave me the freedom to explore on my own and provided me support in every step. Thank you, Alex, for showing me what it means to be passionately curious. Thank you, Ali, for teaching me how to ground my observations in mathematics.

Furthermore I wish to dearly thank my two PhD mentors, Javad Shafiee and Ershad Banijamali. Thank you, Javad, for your unwaivering support and friendship – I will not forget all the times you lent me your ear and helped me in discovering my life path and purpose. Thank you, Ershad, for the wonderful 2-week sprint we pulled off together for the NIPS conference. I hope to continue working with both of you in the future.

I would also like to thank Audrey Chung, Robert Amelard, Kaylen Pfisterer for teaching me how to communicate professionally, Hamed Shahsavan and Devinder Kumar for inspiring me to dream bigger, and Francis Li and Avery Ma for being my late-night and weekend companions when there was nobody else at the lab.

Finally, I would like to say that none of this would have been possible without my wife and family. Your relentless support of my ambitions only fueled them more. I would not be here today without you.

Dedication

This is dedicated to the one I love, Fatemeh. Thank you for putting up with my late nights and early mornings.

Table of Contents

List of Tables

List of Figures

Notation

In the derivations below, scalar values are represented using lower-case italics, e.g., $v_1, v_2, ..., v_n$, vectors are represented using bold lower-case notation, e.g, $\boldsymbol{v} \in \mathbb{R}^n$, and matrices are represented using upper-case italics, e.g., $V \in \mathbb{R}^{m \times n}$.

Indexing matrix elements are as follows:

R: a matrix

$R_{i.}$: the i^{th} column of a matrix

$R_{.j}$: the j^{th} row of a matrix

R_{ij}: the coefficient of the matrix at the i, j^{th} index

Chapter 1

Introduction

Many information processing systems and intelligent decision-making systems operate on measured real-world data that often have a large number of components and high dimensionality. To adequately and efficiently handle this sort of data, these system may first obtain lower-dimensional representations of the data samples. As a result, dimensionality reduction enables, among others, data compression, data visualization, machine learning, and handling of large volumes of high-dimensional data enabling researchers across a variety of fields to overcome the *curse of dimensionality* that comes with having more information.

1.1 Traditional Dimensionality Reduction Methods

Methods for dimensionality reduction are plentiful and have been successfully applied to applications such as head pose estimation [60], visualization of biomedical data [50], face [45] and speech recognition [51], and gene expression analysis [78] among others. Different techniques are used across various data setups taking into account assumptions about the complexity and degrees of freedom of the input data, and performances and running-time complexities vary based on the desired level of accuracy and assumptions made about the underlying manifold.

Common methods for dimensionality reduction include Principal Component Analysis (PCA) [77] that finds the optimal embedding with maximum variance, Multi-dimensional Scaling (MDS) [74] that optimizes an eigenvalue problem to find an embedding that preserves pair-wise Euclidean distances, and Isomap [73] which takes the distribution of neighboring points into account in finding an embedding that preserves pair-wise geodesic distances. Along the lines of Isomap, Locally Linear Embedding (LLE) [61] preserves local

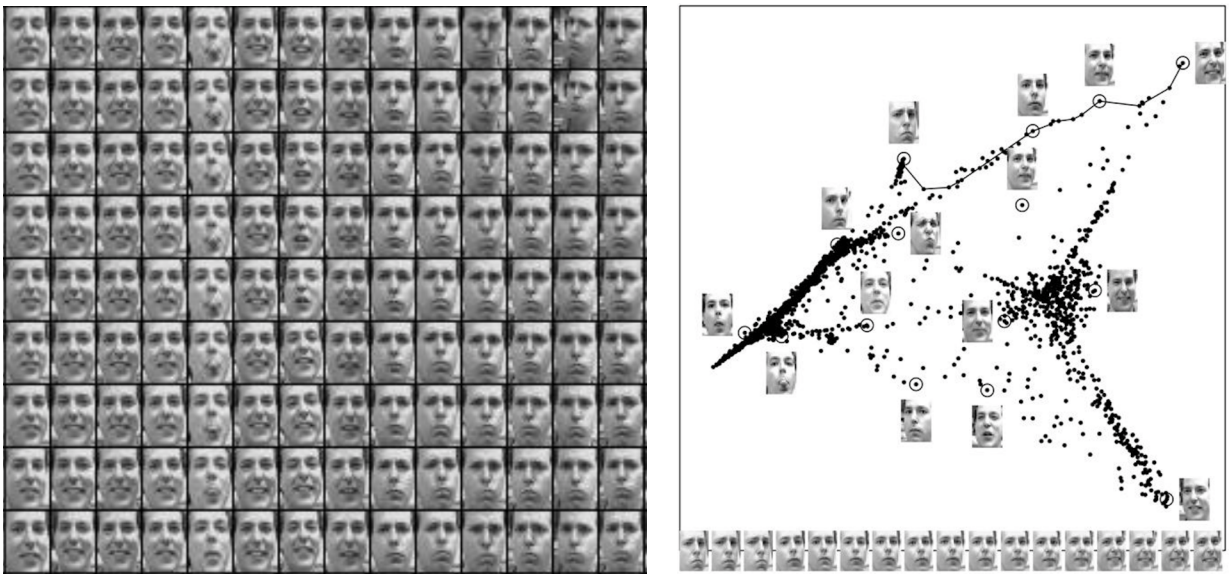


Figure 1.1: Examples of dimensionality reduction applied to high-dimensional data. This Faces dataset was collected by Professor Brendan Frey at the University of Toronto. It consists of 2000 frames of a sequential video capturing his face, where each frame is of size 28×20 pixels, meaning it is 560-dimensional. From looking at this data, it is evident that there are only few degrees of variation in the data, including the angle / orientation of his face, expression, etc., but not for example the relative positioning of the eyes and nose. Therefore, we can maybe find a low (say 2-dimensional) representation of the high 560-dimensional data and use that for further processing. On the right, we see how one method for dimensionality reduction was able to cluster together similar expressions and orientations and plot them on the surface of this page, i.e., in 2 dimensions.

properties of the data manifold by attempting to preserve the reconstruction weights of each sample obtained from writing / reconstructing each original sample as a linear combination of its nearest neighbors in the original high dimensional space.

Despite the success of these methods, special care must be considered when choosing an appropriate dimensionality reduction method. Specifically, the dependence on data often leads researchers to experiment with multiple dimensionality reduction methods before moving on to the rest of their algorithms. The challenge with running multiple experiments to settle on an appropriate method is further exacerbated when dealing with high dimensional data or a large number of datapoints. In PCA, for example, computing the covariance matrix for a large number of features becomes exponentially more expensive as

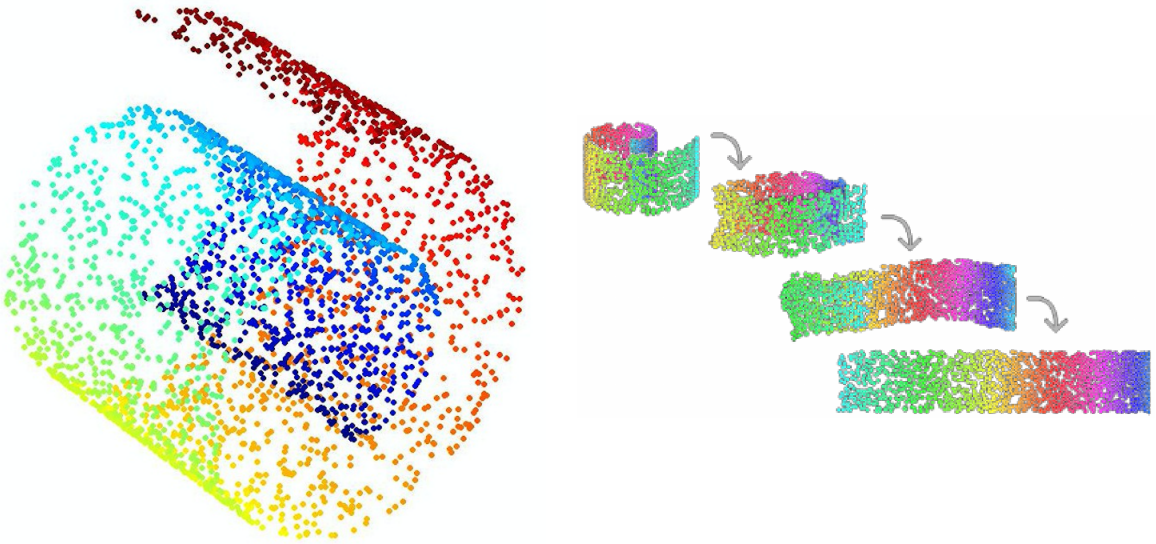


Figure 1.2: Examples of dimensionality reduction applied to complex manifolds. Many times datasets are generated by, or can be best-explained by, complex underlying processes. For instance, inn this Swiss-Roll example, dimensionality reduction helps us by unfolding this manifold through a process called "Manifold Discovery" to explain the locality and geometry of the space. Here we can see that after the unfolding process all blue points remain close together, and all yellow points remain clustered together.

the dimensionality increases, and in MDS or Kernel PCA, constructing pairwise distances to feed into the optimization problem grows exponentially with the number of samples and causes an efficiency bottleneck. As information processing systems tackle larger-scale applications, big-data scenarios are becoming the norm and there seems to be a more urgent need to explore more efficient methods for dimensionality reduction that are universal in their applicability to different datasets.

1.2 Random Projections and the Johnson-Lindenstrauss Lemma

In contrast to traditional methods for dimensionality reduction, *Random Projections* (RP) [14] are simple, efficient, and data-independent methods for dimensionality reduction. The Johnson-Lindenstrauss (JL) theorem [39] asserts that, using a linear projection that is in-

dependent of the samples themselves, one can find an embedding in $\mathcal{O}(\log n/\epsilon^2)$ dimensions where n is the number of samples and ϵ is the error tolerance. Assuming the embedding satisfies a minimum projected space dimensionality of k, this embedding will preserve pair-wise Euclidean distances with high probability. As we shall review in the sections that follow (full proof in Appendix A), this lower-bound depends only on the number of samples n and the error margin ϵ, but not on the original data dimensionality d, rendering random projections as an exceptionally powerful dimensionality reduction tool for very high dimensional data. The simplicity and universal applicability of random projections are further brought to light when one considers how to construct linear random projections: all entries of a $k \times d$ projection matrix can be populated uniformly and independently from a standard Normal distribution [15, 36], or can be independently drawn from $\{-1, 0, +1\}$ [1] resulting in sparse, and consequently more efficient, random projections.

While methods such as linear random projections have proven to be simple and highly efficient in this regard, however, there is limited theoretical and experimental analysis for nonlinear random projections. Recently a study demonstrated that the theory for linear random projections can be extended to nonlinear random projections by applying the Recitifed Linear Unit (ReLU) activation function elementwise on the embedding [25]. The authors demonstrate that this form of nonlinear random projection performs a class-aware embedding where the embedding places objects of the same class closer to one another after the projection compared to objects of different classes.

1.3 Thesis Contributions

Thus, the general motivation of this thesis is designing new forms of random projection for prediction and dimensionality reduction in big-data domains where traditional methods suffer from exponentially more expensive compute power required. These domains include classification of high-dimensional but scarce data samples, and dimensionality reduction of high-dimensional data. In this regard, the contributions of the present thesis are as follows: (i) proposes two new forms of linear random projections inspired by biology, and comprehensively tests these setups on standard imaging datasets and against standard linear random projections, (ii) proposes a new form of nonlinear random projection setup called *ensemble of nonlinear maximum random projections* along with extensive emperical tests on standard imaging datasets and against standard dimensionality reduction methods, and (iii) provides a self-contained proof derivation of the JL Lemma for Gaussian random matrices to build the theoretical foundation of the thesis.

1.4 Thesis Layout

In Chapter 3, we explore an alternative method for linear random projections that are inspired by distributions of biological synapses observed in the human visual cortex and study its application for classifying high-dimensional datasets with few samples. Later in Chapter 4, we build on recent ideas for nonlinear random projections and introduce *ensemble of nonlinear maximum random projections* as a new form of random projection that rival the performance of data-dependent dimensionality reduction setups while incurring a fraction of the computational cost. Finally in Chapter 5.2.3, we present *Supervised Random Projections* and posit it as a tractable approximate method for supervised dimensionality reduction on large datasets.

Chapter 2

Background

In this chapter, background information for the thesis and related works for each proposed contribution are presented. In Section 2.1, we review the Johnson-Lindenstrauss lemma, the fundamental theory supporting random projections. In Section 2.2 the various forms of random projections are described with a comparative overview of scenarios suited for each. Finally in Section 2.3, relevant theory from Compressed Sensing, Locality Sensitive Hashing, and Random-Weighted Neural Networks are presented.

2.1 Johnson-Lindenstrauss Lemma

The Johnson-Lindenstrauss (JL) lemma is concerned with the following problem. We are given a point-set $\mathcal{P}$ of n points in $\mathbb{R}^d$ where d is typically large. We would like to embed these points into a lower-dimensional Euclidean subspace $\mathbb{R}^k$ while approximately preserving the geometry of the space and relative positioning of all pairs of points. Formally,

Theorem 1. *For any point-set* $\mathcal{P} = \{\boldsymbol{x_1}, \boldsymbol{x_2}, \cdots, \boldsymbol{x_n}\} \subset \mathbb{R}^d$*, any integer* n *(number of samples), and any* $0 < \epsilon < 1$ *(error tolerance), let* k *be a positive integer satisfying*

$$k \geq 4(\epsilon^2/2 - \epsilon^3/3)^{-1} \log n \tag{2.1}$$

then, there exists a map $f : \mathbb{R}^d \to \mathbb{R}^k$ *such that for all* $\boldsymbol{x_i}, \boldsymbol{x_j} \in \mathcal{P}$*, with probability greater than* $1 - \delta$ *we have*

$$(1 - \epsilon)||\boldsymbol{x_i} - \boldsymbol{x_j}||^2 \leq ||f(\boldsymbol{x_i}) - f(\boldsymbol{x_j})||^2 \leq (1 + \epsilon)||\boldsymbol{x_i} - \boldsymbol{x_j}||^2 \tag{2.2}$$

There are many potential benefits of operating in a low-dimensional feature space, including reduced storage space and reduced computational complexity of subsequent algorithms operating in the lower-dimensional space. Take for example the expectation maximization (EM) algorithm that clusters datapoints based on squared intra-class distance. Both run-time and space complexity of the EM algorithm depends exponentially on the dimensionality of the datapoints, and reducing this dimensionality improves the algorithm's performance exponentially.

Over the years, both the statement and the proof of the JL lemma have been sharpened and simplified, while various forms of algorithms have been developed for efficiently constructing JL-embeddings satisfying the conditions of Theorem 1. Achlioptas beautifully chronicles the evolution of these proofs in [1], while studying the works of [15, 36, 23]. One such embedding function $f(\cdot)$ is simply a projection matrix $R \in \mathbb{R}^{d\times k}$ where each coefficient $r_{ij} \sim \frac{1}{\sqrt{k}}\mathcal{N}(0,1) = \mathcal{N}(0, \frac{1}{k})$. Therefore, Equation (2.2) can be written equivalently as:

$$Pr\left[\left|||\boldsymbol{y_i} - \boldsymbol{y_j}||^2 - ||\boldsymbol{x_i} - \boldsymbol{x_j}||^2\right| \geq \epsilon\right] \leq \delta \tag{2.3}$$

where $\boldsymbol{y_i} = R^T\boldsymbol{x_i}$, $\boldsymbol{y_j} = R^T\boldsymbol{x_j}$, and R has coefficients samples independently and identically from certain distributions. One such distribution is the standard normal distribution where $r_{ij} \sim \mathcal{N}(0,1)$, while other distributions are detailed below in Section 2.2. Such *random projection* matrices will henceforth be referred to as RP matrices.

2.1.1 Proof Sketch

In all methods for producing JL-embeddings, the core of the proof revolves around showing that the projection of any vector is sharply concentrated around its expected value. As above, consider a set $\mathcal{P} = \left\{\boldsymbol{x_1}, \boldsymbol{x_2}, \cdots, \boldsymbol{x_n}\right\} \subset \mathbb{R}^d$ with n samples being projected into $\mathbb{R}^k$ where $k \ll d$ and $0 < \epsilon < 1$ is the error tolerance. The setup below depicts linear random projections:

$$\overbrace{\begin{bmatrix} r_{11} & r_{12} & \cdots & \cdots & r_{1d} \\ r_{21} & r_{22} & \cdots & \cdots & r_{2d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r_{k1} & r_{k2} & \cdots & \cdots & r_{kd} \end{bmatrix}}^{R^T}_{k\times d} * \overbrace{\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ x_{d1} & x_{d2} & \cdots & x_{dn} \end{bmatrix}}^{X}_{d\times n} = \overbrace{\begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1n} \\ y_{21} & y_{22} & \cdots & y_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ y_{k1} & y_{k2} & \cdots & y_{kn} \end{bmatrix}}^{Y}_{k\times n}$$

where r_{ij} are drawn from certain distributions (detailed in 2.2), and x_{ij} are the features for the datapoints provided. We shall represent each datapoint in the point-set $\mathcal{P}$ as

$\boldsymbol{x_i} = [x_{1i}, x_{2i}, \cdots, x_{di}]^T \ \forall \ i \in [n]$ which collectively populate the columns of the data matrix X. Therefore, X represents the original data and Y represents the embedded data.

In order for such embeddings to be information-preserving, we must satisfy the conditions of Theorem 1 and have a minimum dimensionality of $k \geq 4(\epsilon^2/2 - \epsilon^3/3)^{-1} \log n$. The proof sketch is as follows:

1. Show for fixed $\boldsymbol{x_1}$, $\boldsymbol{x_2}$: $\mathbb{E}[||f(\boldsymbol{x_1}) - f(\boldsymbol{x_2})||_2^2] = ||\boldsymbol{x_1} - \boldsymbol{x_2}||_2^2$
2. Show variance is bounded and small for $\boldsymbol{x_1}$, $\boldsymbol{x_2}$
3. Using Bonferroni's Union Bound, bound the failure probability for all pairs of points

For a detailed proof, refer to Appendix A. The important take-away from this proof is that nothing is fundamental about using Gaussian random projections in particular. Many distributions with unit variance and certain boundedness properties (or higher order moment conditions) suffice and satisfy the conditions of the Johnson-Lindenstrauss lemma. Below, we shall study the various forms of random projections that provide a computationally simple yet exceptionally powerful method for embedding data in lower dimensional space while preserving the information in the point-set.

2.2 Various Forms of Random Projections

2.2.1 Linear Random Projections

In Section 2.1.1 we saw how that many family of distributions have properties (i.e., unit variance and certain boundedness properties) that satisfy the conditions for information preserving embeddings of random projections. In this section, we briefly touch on some of the most common distributions, as detailed in Table 2.1. While Gaussian random projections are the most common in the literature and easiest to derive bounds for, applying this form of random projection requires multiplying out a dense matrix of floating point numbers by the intended data matrix. This creates a two-fold challenge: i) random projections aim to design the simplest embedding strategy, and taking the product of a dense matrix with a data matrix (which is naturally large because of operating in a big-data domain) is computationally expensive, and ii) random projections are often used directly on the sensors that collect the data (see Compressed Sensing); these sensors are often costly (mainly why compressed sensing is used in the first place) and have limited computational

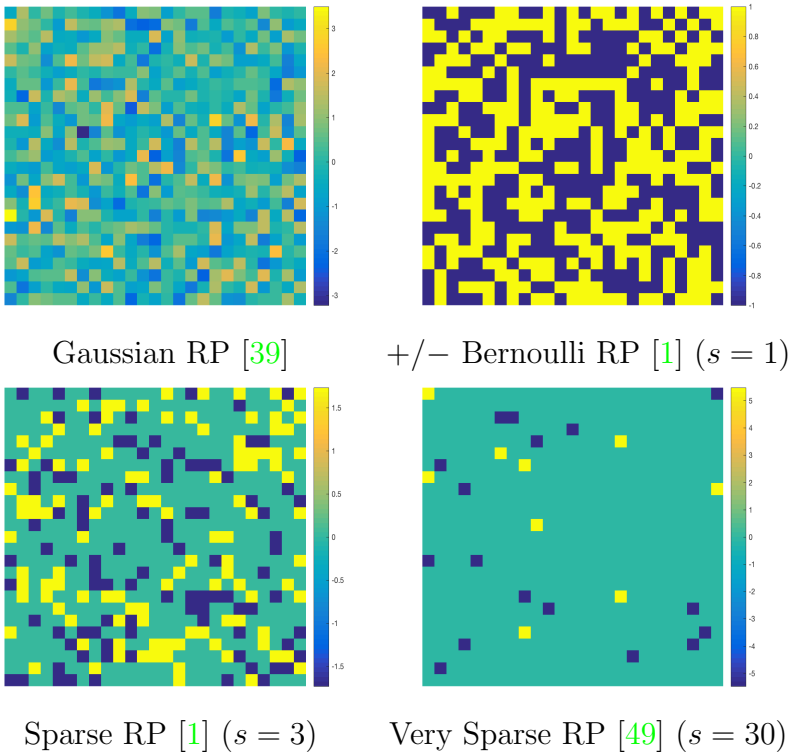


Figure 2.1: Different forms of 25×25 linear random projections with varying sparsity (in brackets). Note that the form of Very Sparse RP presented in Table 2.1 can be used to generate both $+/-$ Bernoulli RP ($s = 1$) and Sparse RP ($s = 3$). Green areas of the projection matrix correspond to coefficients with magnitude 0 which when multiplied by the data matrix do not constitute an add-multiply operation and hence are faster to execute.

capacity; therefore designing random projection schemes that don't involve floating point operations is important.

Overcoming these two challenges is the general motivation of newer forms of linear random projections. In [1], the $+/-$ Bernoulli RP and Sparse RP schemes were put forth, and in recent, [49] proposed Very Sparse RP that push the limits of the earlier method even further. As depicted in Figure 2.1, these newer forms of random projections reduce the number of operations required for stable random projections by a factor of 100 or more compared to Gaussian random projections. Proofs and bound derivations for these methods, albeit more involved, can be found in the corresponding references.

Table 2.1: Various family of distributions for linear random projections.

Gaussian [39]	+/− Bernoulli [1]	Sparse [1]	Very Sparse [49]
$r_{ij} \sim \mathcal{N}(0,1)$	$r_{ij} \sim \begin{cases} +1 & w.p.\ \frac{1}{2} \\ -1 & w.p.\ \frac{1}{2} \end{cases}$	$r_{ij} \sim \sqrt{3} \begin{cases} +1 & w.p.\ \frac{1}{6} \\ 0 & w.p.\ \frac{2}{3} \\ -1 & w.p.\ \frac{1}{6} \end{cases}$	$r_{ij} \sim \sqrt{s} \begin{cases} +1 & w.p.\ \frac{1}{2s} \\ 0 & w.p.\ 1 - \frac{1}{s} \\ -1 & w.p.\ \frac{1}{2s} \end{cases}$

2.2.2 Nonlinear Random Projections

While linear random projections have been studied extensively in the literature, recent research has studied and formulized the effect of nonlinearities on the shape of embedding space. Notably, Giryes *et al.* [25] postulate that element-wise rectification of randomly projected point sets are better suited for prediction tasks such as classification. In particular, the authors show that nonlinear rectified RP activation functions will perform class-aware embedding of the data that is sensitive to angles between points: *such embeddings tend to decrease the Euclidean distances between points with a small angle between them ("same class") more than the distances between points with large angles between them ("different classes").* Nonlinear random projections will be explored in detail in Chapter 4.

2.3 Relevant Work & Applications

2.3.1 Compressed Sensing

Compressed Sensing [6, 18] is a signal processing technique concerned with the problem of efficiently acquiring and reconstructing a signal. A wide variety of signals, including audio and natural images are very high dimensional and can be costly to acquire. However, these signals are often k-sparse and or compressible and therefore can be well-approximated by a linear combination of a few atoms of some redundant dictionary [52, 7, 54]. The sparsity of these signals allows for approximate recovery even when only a small number of nonadaptive [1] linear measurements (including random projections) have been made of the the data. This setup is very similar to that of random projections seen above. Here, $\boldsymbol{y} \in \mathbb{R}^k$ is the information we have obtained from the $k-$sparse signal $\boldsymbol{x} \in \mathbb{R}^n$, $k \ll n$ via $\boldsymbol{y} = \Phi\boldsymbol{x}$, where the encoder $\Phi \in \mathbb{R}^{k \times n}$ has its coefficients independently sampled from certain distribution functions. To extract the information contained in $\boldsymbol{y}$ regarding $\boldsymbol{x}$, a decoder Δ maps from $\mathbb{R}^k$ back to $\mathbb{R}^n$, and is designed to provide an approximation $\tilde{\boldsymbol{x}} \coloneqq \Delta(\boldsymbol{y}) = \Delta(\Phi\boldsymbol{x})$

[1] order-invariant

to $\boldsymbol{x}$. This mapping is typically nonlinear, and finding good encoder-decoder pairs is the central question in Compressed Sensing. Properties of the sensing matrix Φ such as the restricted isometry property [5] determine the distributions that generate good encoders based on the likelihood of small error if the decoder is forced to reconstruct a k-sparse signal. Therefore, the design of data-agnostic encoders for Compressed Sensing is based on the same theory for random projections.

2.3.2 Locality Sensitive Hashing

In the field of information retrieval, the use of random projections was central to the embedding of very high dimensional data for applications such as Locality Sensitive Hashing (LSH). LSH provides an efficient approximate solution to the Nearest Neighbour Search (NNS) problem, which is a problem defined by a collection of objects (represented by a set of features in a high-dimensional attribute space) and a query object with the same features represented in the same high-dimensional space. In the Nearest Neighbour Search problem we are interested in finding an object (or k objects) in the collection that are most similar to the query object. This problem has been well studied and efficient algorithms have been proposed for objects with low-dimensional representations in Euclidean space $\mathbb{R}^d$ under some l_p norm, however, many practical problems operate on high-dimensional data ranging anywhere from 10s to 1000s of features. Such applications include data compression [24], databases and data mining [28], information retrieval [17, 21, 63], machine learning [11], and pattern recognition [12, 19].

The random projection scheme used in LSH was capable of overcoming the curse of dimensionality for efficiently addressing the Approximate Nearest Neighbor Search (ϵ-NNS) problem [2]. In their work, A. Andoni and P. Indyk demonstrated that the randomness inherent to hashes resulted in an algorithm that doesn't guarantee an exact answer, but instead provided a high probability guarantee of finding the nearest neighbor, while benefiting from having to search in a much lower dimensional space. The key idea here was that upon randomly projecting a collection of objects through several hash functions (with certain structural properties such as orthonormality), objects which are close to each other in the input space have a higher probability of collision in the projection space, compared to objects that are more distant in the input space. This in turn results in faster and more accurate nearest neighbor lookup.

2.3.3 Random-Weighted Neural Networks

Extreme Learning Machines

Extreme Learning Machines (ELMs) have recently emerged as an alternative learning strategy to standard feedforward neural networks. Initially proposed for regression in single hidden-layer feed-forward neural networks [34], ELMs have since been extended to multi-layered networks [72], and have been applied for prediction (classification and regression [33]), feature selection [44], as well as dimensionality reduction [43]. Unlike the back-propogation (BP) algorithm typically employed in feed-forward neural networks, ELMs enforce random weights for all-but-final layer of the network, and reduce learning to analytically solving the following linear system of equations:

$$H\boldsymbol{\beta} = T$$

where H is the data matrix containing all data points projected to the penultimate network layer, $\boldsymbol{\beta}$ are the weights between the penultimate and final network layer, and T are the corresponding target outputs of the network for the input data. The claim is that even random weighted layers with nonlinear activations have universal approximation characteristics [31]. Experimental results with ELMs [32] demonstrate massive improvements in training time and competitive generalization performance compared to their BP-equipped counterparts. Random projections and ELM theory are similar in that both use random weights to project into a space with desirable properties. In this work, however, we shall show that random projections for prediction and dimensionality reduction can forgo the need for analytically solving the system of equations above. This posits random projections as a very simple and universally applicable solution for prediction and dimensionality reduction.

Random-weighted Deep Neural Networks

Initially observed by Jarrett et al. [38], and later extended by others to various architectures and for various applications [58, 65, 13, 9], Deep Neural Network (DNN) architectures with fixed (i.e., untrained) random weights have been surprisingly successful in supervised and unsupervised classification. Jarrett et al. experimentally showed that the classification error rate of a 1- or 2-layer convolutional neural network (CNN) with fixed random weights is comparable to that of the network trained in supervised mode for small training sets such as the Caltech 101 [22] or medical imaging datasets.

Saxe et al. [65] demonstrated how the use of random weights can act as a proxy for rapid prototyping of deep neural architectures by separately attributing network performance to both the architecture, and the optimized / trained weights.

Pinto et al. [58] extended the use of randomness beyond kernel weights and into a broader set of neural network parameters and hyperparameters. In their work, the architecture, learning rate, number of neurons, as well as the kernel weights themselves were either randomly chosen from a pre-defined range of categorical possibilities, or independently sampled from a 1-D Gaussian. In that work and in subsequent work [13] the same group demonstrated state-of-the-art performance on a multitude of basic object recognition tasks by *blending* the top performing models discovered using this search strategy.

Chung et al. [9] benchmarked the use of random projection kernels against various kernel approximation methods, and demonstrated that a relatively shallow network with 3 layers outperforms state-of-the-art kernel approximations with the same number of basis expansions.

We encourage the reader to note a crucial difference between the of previous authors and that which is presented in this thesis. In this work, we demonstrate that any network equipped with the proposed generated random kernels can perform really well, forgoing the need for searching over a large parameter space to find optimal configurations. Furthermore, the architectures presented in this section are similar to that of ELMs presented above: both are comprised of one or more random layers followed by a learned stage which adjusts weights analytically, or through back-propogation. In this thesis, we explore random projection setups (shallow and deep) followed by a much simpler learning paradigm: 1-Nearest Neighbors. This allows us to decouple the effect of random projections and final layer training, and sheds light on the geometry of the embedding through random projections itself. A good embedding is one that does not depend on powerful machine learning algorithms to work, and we beleive random projections offer that.

Chapter 3

Biologically Inspired Linear Random Projections

In the previous chapter, we discussed various forms of linear random projections and compared them in terms of efficiency, applicability, and ease of proof. This inspired us to search for new forms of distributions that can provide similar embedding guarantees for specific applications. One such application area is multi-class classification of imaging data, as many recent machine learning algorithms are benchmarked on such datasets. In designing new random projection setups, we were inspired by synaptic weight distributions in the human cortex which empowers the human visual system, the source of inspiration for many new machine learning architectures such as convolutional neural network (CNN). This chapter outlines our process and the design of these new forms of linear random projections, inspired by biology. In Section 3.1 we review the background and relevant literature detailing synaptic weight formation in the human brain. Section 3.2 proposes two new linear random projection setups and Section 3.3 comprehensively tests these setups on standard imaging datasets and against standard linear random projections. We summarize our findings and discuss future work in Section 3.4.

3.1 Introduction

We have recently witnessed an explosive growth in machine learning research focused on modelling and real-world inference problems. Notably, deep learning models such as deep neural networks (DNN) are a particularly powerful and biologically inspired class of learning algorithms that have consistently demonstrated state-of-the-art performance on tasks

such as object recognition, image classification, image segmentation, and speech recognition. A particular type of DNN that has proven to be very effective in recent year are convolutional neural networks (CNNs) (see [35]) which are architecturally made up of layers of neurons modelled after simple and complex cells in the visual cortex.

In order to train a DNN for a prediction task such as classification, the synaptic strengths of the network are optimized based on training data. Optimizing a large-scale artificial neural architecture such as a CNN for classification in a generalizable manner, however, requires a large number of input image samples. This may be prohibitive in many practical scenarios where labeled data is limited. To ameliorate this dependence, we explore whether it is possible to sidestep the training of a large portion of learnable parameters—synaptic strengths—in a neural network. More particularly, we are motivated by [20] where strong modelling and inference performance was exhibited when random synaptic strengths are leveraged in modelling of functional brain computationally. This suggests that the inherent structure of deep neural networks may itself be enough to elicit a powerful modelling and inference performance even when the formation of synaptic strengths are random.

In particular, we draw inspiration from a number of studies that investigated the distribution of synaptic strengths in the biological brain. For example, it has been observed that the synaptic strengths of certain synapses such as the excitatory synapses can be well modelled as random variables following well-known distributions such as truncated Gaussians [3]. Furthermore, Song *et al.* [71] found that the underlying synaptic strengths follows a log-normal distributions. Other studies [55, 8] suggested a correlated relationship between synaptic strengths in earlier layers of the visual cortex, specifically circular concentric receptive fields modelled after Lateral Geniculate (LGN) cells.

Inspired by the aforementioned observations [71, 55, 8], we perform an exploratory study on different uncorrelated and correlated probabilistic generative models for synaptic strength formation in deep neural networks and the potential influence of different distributions on modelling performance particularly for the scenario associated with small data sets.

3.2 Methodology

Here we model the synaptic strength distribution of the deep neural network as $P(\mathcal{W})$ where $\mathcal{W}$ is the set of synaptic strengths $\mathcal{W} = \left\{w_i\right\}_1^n$ and n is the number of synapses. In order to explore the effect of different probabilistic generative models for synaptic formation on modelling and inference performance in a focused manner, in this study we

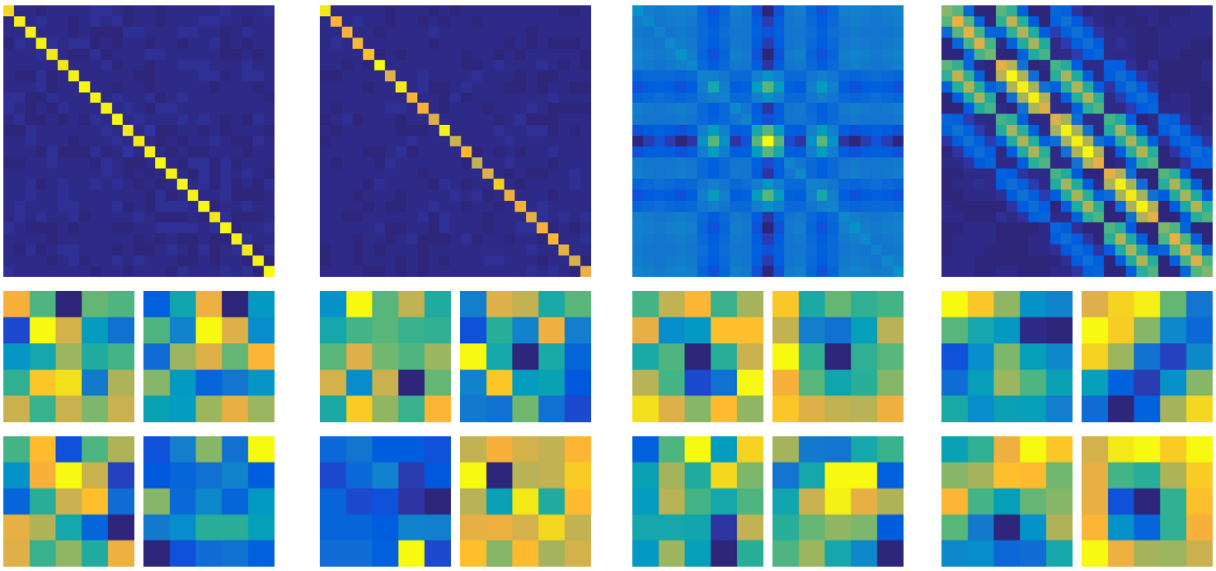


Figure 3.1: Examples of various CNN kernels. The top row depicts the 25×25 covariance matrix that generated each of the four 25-dimensional sample kernels below it. From left to right we see a normal Gaussian distribution, a Log-Normal distribution, Center-Surround kernels, and Gaussian blurred kernels.

restrict the network architecture to be a convolutional neural network (CNN) architecture. More specifically, the synaptic strengths in the convolutional layers are synthesized based on $P(\mathcal{W})$ and are not fine-tuned, whereas the synaptic strengths of fully connected layers are synthesized and then trained to reach their complete modelling capabilities. This setup allows us to localize the effect of $P(\mathcal{W})$ on synaptic strengths and fairly compare the modelling and inference performance of different synaptic formation drawn from various underlying biologically-inspired probability distributions. Furthermore, each random variable corresponding to a synaptic strength denoted as w_i are drawn from a probabilistic generative model $P(\mathcal{W})$. In this study, we explore three different distribution models based on past biological studies:

I Normal Gaussian: $P(\mathcal{W}) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}} \exp(-w_i^2/2)$

II Log-normal: $P(\mathcal{W}) = \prod_{i=1}^{n} \frac{1}{w_i \sigma \sqrt{2\pi}} \exp\left(\frac{\ln(w_i-\mu)^2}{2\sigma^2}\right)$

III Correlated center-surround: $P(\mathcal{W}) = \prod_{i=1}^{n} \frac{1}{\sqrt{|2\pi\Sigma_i|}} \exp(-\frac{w_i^T \Sigma_i w_i}{2})$[1]

Table 3.1: Impact of different probabilistic generative models for synaptic strength generation on modelling performance for 3 small datasets (see text on how datasets were generated). The synaptic strengths of the convolutional layers were generated from distributions describing synaptic strengths in the visual cortex. The convolutional layer synapses are frozen and not trained, whereas the fully connected layers of the CNN are trained over. Accuracy out of %100. Highest performing setups are in bold.

Dataset	Normal	Log-Normal	Center-Surround	Fully Trained
CIFAR-10	$19.83_{\pm\ 00.74}$	$25.67_{\pm\ 00.17}$	$\mathbf{26.40_{\pm\ 00.74}}$	$20.37_{\pm\ 00.62}$
MNIST	$72.52_{\pm\ 00.21}$	$\mathbf{80.89_{\pm\ 00.45}}$	$78.08_{\pm\ 01.19}$	$79.01_{\pm\ 01.39}$
SVHN	$26.40_{\pm\ 00.22}$	$\mathbf{30.32_{\pm\ 00.17}}$	$\mathbf{30.86_{\pm\ 00.57}}$	$27.70_{\pm\ 01.79}$

This approach to synapse strength formation can enable a drastic reduction in the number of parameters that need to be trained, which is an important factor in scenarios with small number of training data.

3.3 Experimental Setup

Followed by biological observations, the effect of three different $P(\mathcal{W})$ are examined on a same convolutional neural network (CNN) architecture here: I) normal Gaussian distribution, II) log-normal Gaussian distribution ($\mu = -0.702$, $\sigma^2 = 0.9355$ from [71]), and III) correlated center-surround distribution.

In order to experiment the effect of different synaptic strength distributions on modelling performance, a CNN is utilized consisting of a convolutional layer comprising of 64 kernels with receptive fields of size 5×5, a max-pooling layer with stride 2, and a rectified nonlinear unit, as well as two fully connected layers inspired by LeNet's fully connected layer architecture [48] and have a $1024N - 64N - 10N$ structure (input - hidden - output).

[1] Σ_i is the covariance matrix at synapse i, where the non-zero off-diagonal elements characterize the correlation between neighboring synapses. This correlated distribution can be thought of as a 2D Gaussian fitler whose center is located in the middle of the receptive field (i.e., convolutional kernel). In our experiments, the coefficient at any point in the 5×5 receptive field was sampled from a noisy 2D Gaussian filter as follows:

$$w_i \sim exp\Big(\frac{-(x - x_0)^2 - (y - y_0)^2}{2\sigma^2}\Big) + \lambda\, \mathcal{N}(0, 1)$$

In this exploratory study, we examined three standard and publicly available object classification datasets including MNIST hand-written digits [48], Street View House Numbers SVHN [56], and CIFAR-10 object recognition dataset [46] for the scenario of small training datasets. To mimic such a scenario 38 samples per each class label (i.e., 10 class labels for each dataset) were randomly selected from the available training data in each dataset to form a small dataset. However to compute the test accuracy, the models are tested with all available testing samples. The reported results (mean and std) are computed based on three runs.

3.4 Summary

Table 3.1 summarizes the results of our experiments. We also report the classification performance of the same CNN architecture on these datasets where the CNN is completely trained, and all synaptic strengths are fine-tuned. As expected, the small number of training samples (i.e., 38 per class) results in the CNN's relatively poor classification performance, as is evident from the right-most column of Table 3.1 named "Fully Trained".

Interestingly, sampling the convolutional synaptic strengths from a normal Gaussian distribution ("Normal" column) yields a classification performance very similar to that of "Fully Trained". This may suggest that in the scenario with very little data, learning a generalizable classification system may not be worth the effort put into training as the performance is similar to that of random valued convolutional synaptic strengths.

The most surprising of the preliminary results can be seen in the "Log-Normal" and "Center-Surround" columns. One possibility that these results suggest is that sampling the synaptic strengths of a CNN from well-known distributions that model synaptic strengths in the visual cortex can result in a classification system that potentially outperforms carefully fine-tuned CNNs on small datasets. This result is a powerful first step towards designing deep neural networks that do not require many data samples to learn, and can sidestep / reduce the burden of current training procedures while maintaining or boosting classification and modelling performance.

There are a number of exciting avenues of future research in this regard. Firstly, we are excited to explore this same effect on deeper networks with more synapses, and to investigate how and whether these synaptic strength distributions may be used to design more efficient architectures and training algorithms. Another interesting direction is to explore the effect of increasing datasets size on classification performances — we expect that the performance of random-weighted neural networks to break down as we move to operate in larger datasets regimes. Finally, the current results were inspired by synapse

distributions emperically observed in the visual cortex of humans, however, it is important that future research ground these observations in theory of random projections (similar to Appendix A) as such theory may suggest newer forms of random projections and derivations thereof that have not yet been explored.

Chapter 4

Nonlinear Random Projections

In the previous chapters, we took a deep dive into various forms of linear random projections. The methods discussed were all data- and task-agnostic in their construction of random projections, and provided guarantees of geometry preservation in the embedding. Linear methods are attractive due to their simplicity and because the assumption that the data lies near a low-dimensional manifold applies broadly to many datasets. However, as is common in machine learning, nonlinear extensions can provide powerful generalizations of linear methods and potentially provide more appropriate methods for complex natural datasets. Therefore in this chapter, we focus on exploring a new form of nonlinear random projection setup called *ensemble of nonlinear maximum random projections*. In Section 4.1 we outline the motivation of this chapter. In Section 4.2 we review the background and relevant studies detailing current nonlinear random projection setups and detail the aforementioned new setup. Section 4.3 comprehensively tests this setup on standard imaging datasets and against standard dimensionality reduction methods. Our findings are discussed in Section 4.4 and summarized in Section 4.5.

4.1 Introduction

In this study, we review the theoretical framework for random projections and nonlinear rectified random projections, and introduce *ensemble of nonlinear maximum random projections*. We empirically evaluate the embedding performance on 3 commonly used natural datasets and compare with linear random projections and traditional techniques such as PCA, highlighting the superior generalization performance and stable embedding of the proposed method.

Nonlinear dimensionality reduction methods such as this hold promise to offer an advantage over their linear counterparts for real-world data, as real-world data is likely to lie on or near a highly nonlinear manifold. This is the question we explore in this study. We extend the line of work above in this experimental study by employing an ensemble of random projections and using the maximum activations as the embedding coefficient. In the work that follows, we empirically demonstrate how this form of random projection leads to stable low-dimensional embeddings that perform better than linear random projections [23], nonlinear rectified random projections [25], and PCA [77].

4.2 Methodology

Inspired to extend linear random projection to nonlinear random projections to tackle complicated real-world datasets, we first review the theory on linear RPs in the context of dimensionality reduction. Dimensionality reduction attempts to find an embedding $Y \subset \mathbb{R}^k$ of the original set $\boldsymbol{X} = \left\{\boldsymbol{x_1}, \boldsymbol{x_2}, \cdots, \boldsymbol{x_n}\right\} \subset \mathbb{R}^d$. In particular, dimensionality reduction based on random projections rely on the Johnson-Lindenstrauss (JL) theorem [39] to assert the existence of an embedding that preserves all pair-wise Euclidean (l_2) distance, with high probability. More specifically,

Theorem 2.1 *For any set* $\boldsymbol{X} = \left\{\boldsymbol{x_1}, \boldsymbol{x_2}, \cdots, \boldsymbol{x_n}\right\} \subset \mathbb{R}^d$*, any integer* n *(number of samples), and any* $0 < \epsilon < 1$ *(error tolerance), let* k *be a positive integer satisfying*

$$k \geq 4(\epsilon^2/2 - \epsilon^3/3)^{-1} \log n \tag{4.1}$$

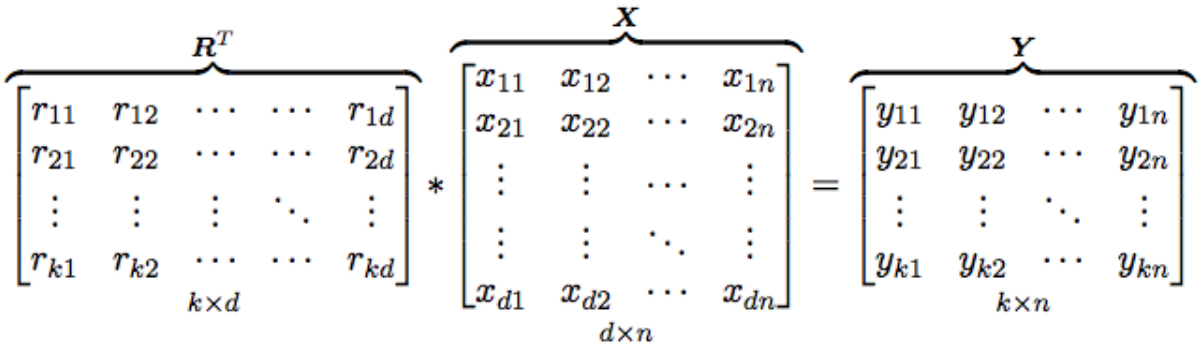
then, there exists a map $f : \mathbb{R}^d \to \mathbb{R}^k$ such that for all $\boldsymbol{x_i}, \boldsymbol{x_j} \in X$, with probability greater than $1 - \delta$ we have

$$(1 - \epsilon)||\boldsymbol{x_i} - \boldsymbol{x_j}||^2 \leq ||f(\boldsymbol{x_i}) - f(\boldsymbol{x_j})||^2 \leq (1 + \epsilon)||\boldsymbol{x_i} - \boldsymbol{x_j}||^2 \tag{4.2}$$

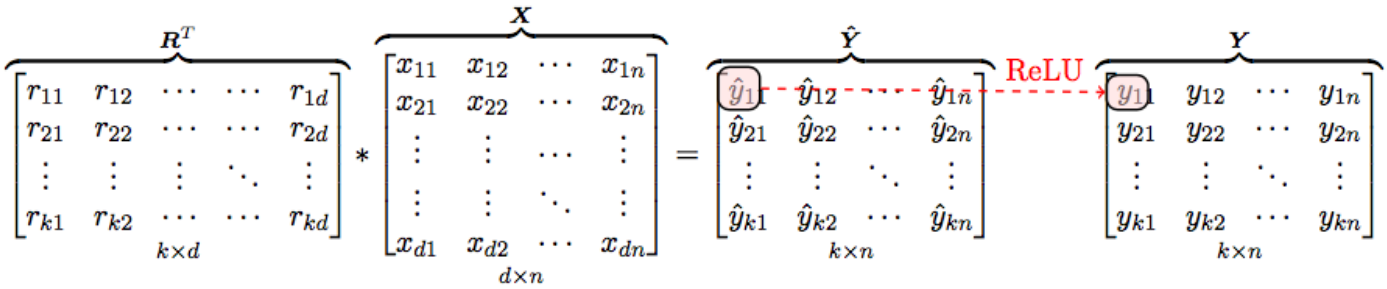
One such embedding function $f(\cdot)$ is simply a projection matrix $R \in \mathbb{R}^{d \times k}$ where each coefficient $r_{ij} \sim \frac{1}{\sqrt{k}}\mathcal{N}(0, 1) = \mathcal{N}(0, \frac{1}{k})$. Therefore, the above can be written equivalently as

$$Pr\left[\left|||\boldsymbol{y_i} - \boldsymbol{y_j}||^2 - ||\boldsymbol{x_i} - \boldsymbol{x_j}||^2\right| \geq \epsilon\right] \leq \delta \tag{4.3}$$

where $\boldsymbol{y_i} = R^T\boldsymbol{x_i}$ and $\boldsymbol{y_j} = R^T\boldsymbol{x_j}$. For a proof of the above theorem, as well as other forms of embeddings, refer to [15, 36, 1]. Linear random projections are depicted as follows:



Giryes *et al.* [25] extended this line of work to nonlinear RPs by applying an activation function on the embedded samples, $\boldsymbol{y_i}\ \forall\ i \in [n]$. In particular, a ReLU operator ($\rho(w) = w \cdot \mathbb{1}\{w \geq 0\}$) was applied element-wise to each coefficient of the embedded samples, as depicted below:



This resulted in the introduction of an additional term in (2.3) which depends on the angular distance between samples in the original space (i.e., the $\boldsymbol{x_i}s$)

$$Pr\left[\left|||\boldsymbol{y_i} - \boldsymbol{y_j}||^2 - \left(\frac{1}{2}||\boldsymbol{x_i} - \boldsymbol{x_j}||^2 + ||\boldsymbol{x_i}||\ ||\boldsymbol{x_j}||\ \Psi(\boldsymbol{x_i}, \boldsymbol{x_j})\right)\right| \geq \epsilon\right] \leq \delta \tag{4.4}$$

where $\boldsymbol{y_i} = \rho(R^T\boldsymbol{x_i})$, $\boldsymbol{y_j} = \rho(R^T\boldsymbol{x_j})$, $\Psi(\boldsymbol{x_i}, \boldsymbol{x_j}) = \frac{1}{\pi}\big(\sin(\theta) - \theta\cos(\theta)\big)$, and $\theta = \angle(\boldsymbol{x_i}, \boldsymbol{x_j})$, the angular distance between $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$. The authors show that $\Psi(\boldsymbol{x_i}, \boldsymbol{x_j})$ is approximately equal to $0.5(1 - \cos(\theta))$, helping us understand that the probability bound (4.4) suggests that nonlinear rectified RPs activation function will perform class-aware embedding of the data that is sensitive to angles between points: *such embeddings tend to decrease the Euclidean distances between points with a small angle between them ("same class") more than the distances between points with large angles between them ("different classes").*

In addition to ReLU as an activation function for nonlinear RPs, the authors of [25] claim a similar analysis can be derived for the spatial pooling operation commonly used in

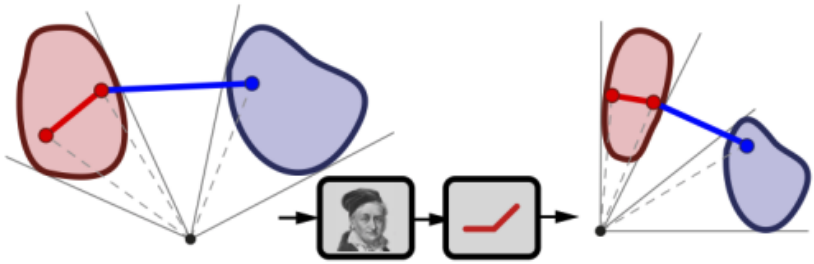
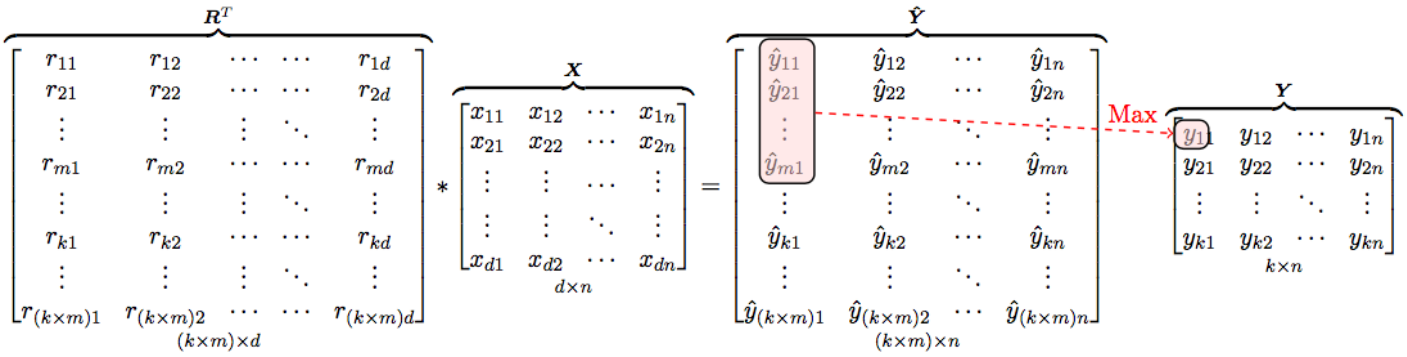


Figure 4.1: Effect of Nonlinear Rectified Random Projections on intra- and inter-class Euclidean distances. The distance between the blue and red points shrinks less than the distance between the red points as the angle between the latter is smaller. Source: [25].

convolutional neural networks (CNNs). In this work, we explore the effect of choosing the max activation as the embedded feature on the quality of the embeddings for the application of dimensionality reduction. In contrast to spatial pooling used in CNNs, that supports embedding-robustness via spatial invariance, our strategy selects the maximum activation of m *randomly* selected features as the embedded coefficient. This form of nonlinearity is supported by an ensemble of random projection matrices $\{ R^{(1)}, \cdots, R^{(m)}\} \subset \mathbb{R}^{d \times k}$ that embed each input sample $\boldsymbol{x_i}$ into $\boldsymbol{y_i}$ via

$$\boldsymbol{y_{ij}} = \max \left\{ (\boldsymbol{R_j}^{(1)})^T \boldsymbol{x_i}, \cdots, (\boldsymbol{R_j}^{(m)})^T \boldsymbol{x_i} \right\} \; \forall \; j \in [k] \tag{4.5}$$

where $\boldsymbol{y_{ij}}$ is the j^{th} coefficient of embedded point $\boldsymbol{y_i}$, and $\boldsymbol{R_j}^{(l)}$ is the j^{th} column of the l^{th} projection matrix $R^{(l)}$. We refer to this method as *Ensemble of Nonlinear Maximum Random Projections*, as depicted below:



4.3 Experimental Setup

In this section, we describe our experimental setup, parameters, and metrics used to compare the performance of the proposed method against other dimensionality reduction methods. In the experiments below, n is the number of samples, d is the original dimensionality, and k is the projected dimensionality of the embedded space. Table 4.1 summarizes the results averaged over 10 runs.

4.3.1 Evaluation Metric

We assess the quality of the embedding by evaluating how the local structure is retained in the projected space. This is accomplished by measuring the generalization error of 1-nearest neighbor (1-NN) classifier trained on the low-dimensional representation of the data (as is done, e.g., in [64, 75]). Ideally, dimensionality reduction reduces the number of data features while maintaining a certain level of generalization performance. As we shall see, in many cases dimensionality reduction methods lead to improved generalization errors in the lower dimensions, a characteristic much desired.

4.3.2 Datasets

For our experiments, we selected three datasets that represent tasks from a variety of domains: (i) the CIFAR-10 dataset [46], (ii) the STL-10 dataset [10], (iii) the ImageNet Tiny dataset. These datasets were selected because they satisfy the theoretical conditions for stable embedding using random projections[1].

The first two datasets comprise of 10 classes of natural scene images with image size $32 \times 32 \times 3$. This value constitutes the original dimensionality of the samples. The CIFAR-10 and STL-10 datasets contain $50,000$ and $5,000$ training samples, respectively. For computational reasons, we randomly selected 250 samples from each of the 10 classes as the training set, and used the entire test set to compute 1-NN performance.

The final dataset comprised of 200 classes of natural images, with 500 training samples and 50 test samples per class. Each image was the result of a crop of an original image in the ILSVRC dataset [62], where the crop was done using accompanying bounding box information. Even this *tiny* version was prohibitively large to compute 1-NN for, therefore we opted to first resize all cropped images to $32 \times 32 \times 3$, and instead of computing generalization error on the collective of 200 classes, we evaluated performance on 10 randomly selected pairs[2] and computed average performance in each projected dimension. Running

Table 4.1: 1-NN performance of 3 common imaging datasets at different projected dimensions k. All samples have original dimensionality d equal to $32 \times 32 \times 3 = 3072$. Cells in bold demostrate the superior performance of Nonlinear Ensemble RP across various datasets and different projected dimensions $k \geq \mathcal{O}(\log n/\epsilon^2)$. All reported results are the average of 10 runs.

Datasets			Projection Type	Projected Dimensions k						
Name	# Classes	n (training)		64	128	256	512	1024	2048	3072
CIFAR-10	10	2,500	No RP							27.26± 00.00
			PCA	**30.27± 00.00**	**29.28± 00.00**	**27.90± 00.00**	27.40± 00.00	27.30± 00.00	27.25± 00.00	27.26± 00.00
			Linear RP	25.12± 00.32	26.35± 00.37	26.83± 00.43	26.93± 00.27	26.93± 00.26	27.04± 00.13	27.06± 00.18
			Nonlinear Rectified RP	23.05± 00.51	25.32± 00.44	26.33± 00.50	26.79± 00.22	27.08± 00.30	26.98± 00.21	27.12± 00.12
			Nonlinear Ens. Max RP	19.82± 00.44	23.12± 00.47	25.96± 00.47	**27.93± 00.28**	**29.15± 00.25**	**29.82± 00.38**	**30.02± 00.29**
STL-10	10	2,500	No RP							27.56± 00.00
			PCA	**30.52± 00.00**	**30.00± 00.00**	**28.80± 00.00**	28.16± 00.00	27.70± 00.00	27.57± 00.00	27.56± 00.00
			Linear RP	26.58± 00.67	27.36± 00.33	27.64± 00.33	27.69± 00.27	27.68± 00.22	27.69± 00.19	27.69± 00.18
			Nonlinear Rectified RP	25.12± 00.55	26.84± 00.58	27.56± 00.22	27.61± 00.41	27.88± 00.21	27.98± 00.14	27.98± 00.12
			Nonlinear Ens. Max RP	22.90± 00.68	25.60± 00.73	27.51± 00.54	**28.74± 00.45**	**29.46± 00.39**	**29.90± 00.28**	**30.03± 00.31**
ImageNet (Tiny)	2 (avg of 10 pairs)	1,000	No RP							51.50± 00.00
			PCA	51.50± 00.00	51.50± 00.00	51.50± 00.00	51.50± 00.00	**51.50± 00.00**	51.50± 00.00	51.50± 00.00
			Linear RP	51.60± 03.42	50.75± 02.21	51.64± 02.06	**51.64± 01.72**	51.35± 01.23	**51.58± 01.14**	**51.58± 00.88**
			Nonlinear Rectified RP	50.98± 03.56	50.87± 02.48	51.22± 02.13	50.83± 01.71	50.97± 01.27	51.28± 01.00	51.18± 01.10
			Nonlinear Ens. Max RP	**52.53± 03.22**	**51.61± 02.69**	**51.96± 02.17**	50.76± 01.81	51.13± 01.79	51.36± 01.14	51.27± 01.13

1-NN on the entire 200-class dataset resulted in generalization performance of around %2 accuracy for data projected using the proposed method; although the proposed method outperformed other methods, the performances were hardly distinguishable, thus we opted the pairwise strategy.

4.4 Discussion

We first address the surprising result that lower-dimensional representations of the aforementioned datasets often lead to improved generalization performance while having fewer dimensions. A possible explanation for this increase in performance after dimensionality reduction is due to the nature of the datasets, where pixels / features in natural images contain local information that is repeated across neighboring pixels / features. Furthermore, dimensionality reduction is known to make the representations more robust to noise and outliers, potentially leading to improved generalization performance. We note that this trend does not continue to improve performance as we continue to reduce dimensionality further. This is expected because ultimately the dimension is reduced to 1 feature where almost all of the features are discarded and hence we cannot expect superior performance. Results for really low dimensions are not included due to brevity.

We also note that PCA almost always outperforms linear RP, potentially because PCA actively considers the data in its optimization process. Many studies have compared PCA and linear RP. Dasgupta's seminal work [14] studies the tradeoffs between these two methods, demonstrating that although PCA often performs better than linear RP, linear RP enjoys superior time-complexity ($\mathcal{O}(dn)$ for linear RP vs $\mathcal{O}(d^3)$ for PCA). Furthermore, while linear RP can always stably embed the input set into $k = \mathcal{O}(\log n/\epsilon^2)$ dimensions, PCA can at *worst case* embed into $k = \Omega(n)$ dimensions.

Furthermore recall that PCA focuses on solving a *global* minimization (i.e., min $\sum_{i=1}^{n} ||\boldsymbol{x_i} - \boldsymbol{y_i}||^2$ where $\boldsymbol{x_i}$ is the original sample and $\boldsymbol{y_i}$ is the projected sample). This form of optimization does not guarantee that local pairwise distances are preserved. In contrast to PCA, random projections have provably converging bounds on the distortion of local pairwise distances between all pairs of samples in the embedding space with respect to the original space.

[1]To accommodate the theoretical conditions for highly accurate random projections, the embedded space $\mathbb{R}^k$ must satisfy a minimum dimensionality of $\mathcal{O}(\log n/\epsilon^2)$. This minimum dimensionality is > 100 for the number of samples in our experiments.

[2]Sample pairs: (school bus, remote control), (brown bear, german shepherd), (brown bear, lion), (lion, monarch butterfly), (monarch butterfly, steel-arch bridge), $\cdots$

It is also interesting to see the results for nonlinear maximum RP outperform both linear RP and nonlinear rectified RP at higher dimensions. This difference in performance is observed while [25] suggest similar performance for ReLU and spatial pooling activation functions. We hypothesize that the low performance of nonlinear rectified RP in high dimensions may be because roughly half of the features are being set to zero, while the low performance of nonlinear maximum RP in low dimensions may stem from the relatively tighter distribution of features in the embedded space (i.e., (4.5)) and the quality of the l_2 norm in high dimensions. This comparison and reasons for varying performances merit further study.

Furthermore, we observe that the performance of linear RP and nonlinear rectified RP is consistent as the projected dimension k decreases from 3072 to 128 and then drops as k drops from 128 to 64. This suggests that we have crossed the theoretical lower bound for highly stable embedding. In contrast to this, the performance of nonlinear maximum RP follows a gradual decrease in performance as the projected dimension k drops from 3072 to 64.

Finally, a surprising observation in our experiments for ensemble of nonlinear maximum RP was its similar performance to that of ensemble of nonlinear minimum RP, as long as we were consistent in applying the max / min functions on each output dimension. This observation will likely be of importance in future work when deriving theoretical probability bounds for nonlinear maximum RP.

4.5 Summary

In this study, we introduced a new method for nonlinear maximum RPs for stable and class-aware embedding of n data samples from $\mathbb{R}^d$ into $\mathbb{R}^k$. Inspired by theoretical work on linear RPs and nonlinear rectified RPs, and following their stipulation surrounding theory for the proposed method, we perform an experimental study showing the stable and superior embedding of nonlinear maximum RPs compared to prior RP methods on 3 different real-world datasets. Furthermore, we compare the performance of the proposed method with PCA, a commonly used dimensionality reduction technique, and show that the proposed method performs comparatively (in the theoretically allowed range for k) while being computationally much more efficient.

In future work, we would like to derive a theory for the probability bounds of the nonlinear embedding of samples into lower dimensions using the max activation function. Specifically, we would like to assert the claim of [25] that this bound is similar for spatial pooling and ReLU activation functions, and to explore the differences and the interplay

between these two nonlinearities, and variants thereof. Additionally, our preliminary experiments on multi-layered nonlinear RPs (not included here for brevity) hint at the compounded power of such projections in further boosting performance. Theory in this vein is promising for theoretically backing empirical results observed by [38] and [42] where CNNs with random weights competitively performed on classification tasks.

Chapter 5

Supervised Random Projections

Previous chapters have been concerned with positing random projections as an efficient and data-independent method for dimensionality reduction. There, we reviewed standard random projections, presented biologically inspired linear random projections, and attended to ensembles of maximum random projections as extensions of the random projections framework for modeling datasets with few samples and or with nonlinear relationships. This chapter presents one final strategy for random projections, this time suggesting the use of the labels present in the data as a guide for random projections. The presented method is referred to as *Supervised Random Projections*. In Section 5.1, we review the motivation for supervised random projections, and outline the problem we attempt to solve. In Section 5.2, we present relevant work on information maximization between predictor and response random variables, and review the seminal work on kernel approximation relevant to supervised random projection derivation. It is based on these foundations that we are able to derive the formulation for supervised random projections. Section 5.3 then describes our experimental setup including dataset and evaluation metrics. Finally, in Section 5.4 we discuss the results and conclude with future recommendations.

5.1 Introduction

In previous chapters, we have seen random projections posited as a tractable approach for dimensionality reduction for datasets containing many samples or samples with high dimensionality. There, linear random projections were portrayed as transformations that preserve the geometry and relative positioning of the data manifold with high probability, and nonlinear random projections were shown to cluster samples of the same class together

in the embedding space based on the likelihood that samples of the same class have smaller angles between them. While previous random projection setups were data-independent in that the same architecture and distribution were used across different datasets, this chapter suggests an alternative that considers the independent high-dimensional explanatory random variable as well as its corresponding dependent response random variable in the embedding process. Specifically, we seek to devise a random projection scheme where the cross-correlation between projected samples and their labels (in a classification setting) is maximized. We refer to this new setup as *Supervised Random Projections*. In this chapter, we will explore the setup for supervised random projections, and base the derivation in relevant work on Supervised Principal Component Analysis (SPCA), a dimensionality reduction and feature extraction method that has found many applications in data visualization, regression, and classification. While SPCA and its nonlinear extension (i.e., KSPCA) demonstrated highly competitive performance compared to various supervised dimensionality reduction methods [4], the derivation and implementation include a spectral decomposition stage which is a major bottleneck for high-dimensional data (SPCA) or for datasets with many samples (KSPCA). The motivation for this work is to overcome this bottleneck by employing the power of kernel approximation.

5.2 Methodology

In this section, we will review the foundations of Supervised Principal Component Analysis [4], a method to optimally find a projection of our data in a potentially lower-dimensional space that has maximum cross-correlation with the labels. We also present the motivation for, and present an elegantly simple way of, approximating kernel machines, popular methods used broadly in machine learning. Finally, we detail the derivations of the proposed method.

5.2.1 Kernel Approximation

Kernel methods are successful techniques used broadly in many machine learning for problems ranging from classification [67] and regression [80] to metric learning [76, 16] and dimensionality reduction [79, 66]. Kernel methods evaluate a kernel function $k(\boldsymbol{x}, \boldsymbol{y}) = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{y}) \rangle$ for all pairs of training samples and store the evaluated pairwise similarities (defined as inner product between samples in some potentially high-dimensional space [37]) in a *Kernel matrix* [1], K_{ij}, to be used in downstream algorithms. Here, $\phi : \mathcal{X} \to \mathcal{F}$ is

[1] also *Gram matrix*

known as a *lifting* function which maps elements of the observation space $\mathcal{X}$ into a high-dimensional feature space $\mathcal{F}$. The "kernel trick" allows us to evaluate the inner product between any pair of samples in a high-dimensional space by directly plugging in the values of $\boldsymbol{x}$ and $\boldsymbol{y}$ into the closed-form formula for $k(\boldsymbol{x}, \boldsymbol{y})$, essentially evaluating the similarity between these samples in a nonlinear, potentially infinite dimensional space. A commonly used kernel is the RBF kernel: $k(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(||\boldsymbol{x} - \boldsymbol{y}||^2/2\sigma^2\right)$ with variance σ.

Despite the success of these methods, kernel methods have limited applicability in large-scale problems due to poor scaling in the face of increasing number of training samples. This problem presents itself when storing the Kernel matrix, and later when the Kernel matrix is used to evaluate a decision function for a new test sample becomes increasingly computationally expensive as the number of training samples increase. In their seminal work, Rahimi and Recht [59] proposed that mapping the data (both train and test) into a relatively low-dimensional randomized feature space can ameliorate the problem of evaluating the kernel function on all pairs of samples. Their method essentially bypasses the reliance on the implicit lifting function ϕ and instead, uses an explicit mapping that transforms the data from its original space into a low-dimensional Euclidean inner product space using a random feature map $\boldsymbol{z} : \mathbb{R}^d \to \mathbb{R}^k$. This randomized feature map is constructed in a manner ensuring that the inner product between a pair of transformed points approximates their kernel evaluation:

$$k(\boldsymbol{x}, \boldsymbol{y}) = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{y}) \rangle \approx \boldsymbol{\psi}(\boldsymbol{x})^T \boldsymbol{\psi}(\boldsymbol{y}) \tag{5.1}$$

where the parameters of $\boldsymbol{z}$ are random bases sampled independently from the inverse Fourier transform of the desired shift-invariant kernel. Incidentally, this covers a wide class of kernel functions including Gaussian RBF, Laplace, Matern, etc. Thus, instead of evaluating the entries of the kernel matrix individually, the entire kernel matrix K can be approximated via a fixed set of random bases drawn from the above distribution applied to the data samples. This method came to be known as *Random Fourier Features* and was later extended and referred to as *Random Kitchen Sinks*. For a Gaussian RBF kernel with variance σ, one can simply approximate the training data kernel using the following MATLAB code:

```
% N: number of training samples in X
% d: dimensionality of training samples in X
% k: dimensionality of random feature and/or number of random bases
W = randn(k,d) / σ;
b = 2 * pi * rand(k,1);
Ψ = sqrt(1 / k) * cos(W * X + b * ones(1,N));
kernel_approx = Ψ' * Ψ;
```

Inspired by the work above on approximating shift-invariant kernels, other extensions were proposed for various other types of kernels including polynomial kernels [27, 41, 57], and for learning the kernel itself [70]. Parallel to this, while the Random Kitchen Sinks approach presented an efficient way to approximate kernels, it operated in $\mathcal{O}(kd)$ space and time, where k is the expansion/embedded dimension, and d is the input dimension. The FastFood method [47] accelerates Random Kitchen Sinks to $\mathcal{O}(k \log d)$ time complexity and $\mathcal{O}(k)$ storage complexity, making this approach better suited for high-dimensional datasets such as image datasets. While this method would render our results presented in Section 5.3 even more impressive, we decided to base our approximations on the more widely used, and simpler to implement, Random Kitchen Sinks approach.

5.2.2 Maximizing Information Dependence

Suppose we have a dataset $\mathcal{S} = \left\{(\boldsymbol{x_i}, \boldsymbol{y_i})\right\}_{i=1}^{n} \subseteq \mathcal{X} \times \mathcal{Y}$ where $\mathcal{X} \in \mathbb{R}^d$ is the space of all d-dimensional explanatory variables, $\mathcal{Y} \in \mathbb{R}^\ell$ is the space of all ℓ-dimensional response variables. Let $X \in \mathbb{R}^{d \times n}$ and $Y \in \mathbb{R}^{\ell \times n}$ be particular realizations of n random pairs of variables sampled independently from $P_{\mathcal{X},\mathcal{Y}}$. We aim to find an orthogonal projection U of X to maximize the cross-correlation (dependence) between $U^T X$ and Y.

To maximize the dependence of Y on $U^T X$, we turn to the commonly used Hilbert-Schmidt Independence Criterion (HSIC) introduced by Gretton et al. [26]. HSIC is an effective tool for "measuring" (nonlinear) dependence between two random variables. While the exact value of HSIC is measured by computing the cross-covariance between $\mathcal{F}$ and $\mathcal{G}$ (where $\mathcal{F}$ and $\mathcal{G}$ are separable reproducing kernel Hilbert spaces containing all continuous bounded real-valued functions of x from X to R and y from Y to R, respectively), empirical approximations to the HSIC value between random variables X and Y can simply be calculated by evaluating the following on the finite number of available observations

$$HSIC(\mathcal{S}, \mathcal{F}, \mathcal{G}) = \frac{\mathbf{tr}(KHLH)}{(n-1)^2} \tag{5.2}$$

where $H, K, L \in \mathbb{R}^{n \times n}$, $K_{ij} \coloneqq k(\boldsymbol{x_i}, \boldsymbol{x_j})$, $L_{ij} \coloneqq l(\boldsymbol{y_i}, \boldsymbol{y_j})$, and $H_{ij} \coloneqq I - \boldsymbol{e}\boldsymbol{e}^T/n$ is the centering matrix. Therefore, to maximize the dependence between $U^T X$ and Y, K becomes the kernel of $U^T X$ and therefore we must maximize the following

$$\mathbf{tr}(KHLH) = \mathbf{tr}(X^T U U^T X H L H) \tag{5.3}$$

$$= \mathbf{tr}(U^T X H L H X^T U) \tag{5.4}$$

To construct the final optimization problem, we add the condition for orthogonality of the transformation matrix U (incidentally, this condition makes the optimization problem well-defined by bounding the objective function), and we obtain the following:

$$\begin{aligned} \underset{U}{\text{argmax}} \quad & \mathbf{tr}(U^T X H L H X^T U) \\ \textit{subject to} \quad & U^T U = I \end{aligned} \tag{5.5}$$

This optimization problem can solve for U in closed-form. To obtain the final form, and to motivate the randomized approach, we make note of the following facts.

Fact 1) The trace is maximized when $U = [\boldsymbol{u_d}, \boldsymbol{u_{d-1}}, \cdots, \boldsymbol{u_{d-k+1}}]$ contains the eigenvectors corresponding to the top k eigenvalues of the symmetric and real matrix $Q = XHLHX^T$. This can be derived by writing out the Lagrangian form of the optimization problem. Note that when $k = d$, U becomes an orthogonal projection of X into the same space as X, and when $k < d$, U becomes a projection of X into a sub-space of X.

Fact 2) Because Q is a symmetric and real matrix, it can be written as $Q = V\Sigma V^T$ via singular value decomposition (SVD). Here, V is equal to the eigenvectors of $Q^T Q$.

Fact 3) Because Q is a positive semi-definite matrix, the eigenvectors of Q are equal to the eigenvectors of $Q^T Q$.

Combining Fact 2 with Fact 3, we can conclude that V is equal to the eigenvectors of Q in addition to being equal to the eigenvectors of $Q^T Q$ (and QQ^T). Adding this to the intuition gained from Fact 1, we can see that $U = V$ for the optimal solution of the optimization problem above. This means that we can simply find the best projection U of X by decomposing Q via SVD, and taking the first k columns as V (i.e., eigenvectors corresponding to top k singular values of Σ) as the optimal projection matrix. This approach was first presented by Barshan et al. in their work on Supervised Principal Component Analysis (SPCA) [4].

Now that we have established the derivation of SPCA, it is worth noting that exact SVD of $Q \in \mathbb{R}^{n \times n}$ has time complexity $\mathcal{O}(n^3)$ [30]. This is a major obstacle in the face of large-scale data common to today's machine learning problems. As an example, finding an embedding using SPCA for the standard hand-written character classification dataset, MNIST [48], which contains $60,000$ 784-dimensional training samples will approximately require 4.8×10^8 floating-point operations just for decomposition (2×10^{14} for KSPCA). Approximating the SVD by decomposing subsets of Q, instead of Q itself, is one approach to overcome this challenge. An alternative approach is to consider using an approximation to the L kernel present in $Q = XHLHX^T$ in place of the exact L kernel. We shall see how this will allow us to bypass SVD completely.

5.2.3 Supervised Random Projections

The high-level goal of this section is to combine ideas from kernel approximation into the Supervised Principal Component Analysis (SPCA) framework to achieve a randomized approach for SPCA, namely *Supervised Random Projections*. It is noteworthy that PCA is itself a special case of SPCA when labels are either not present, or not used [2]. Therefore, the proposed Supervised Random Projections method is also a novel approach for performing PCA in a randomized manner, namely *Randomized Principal Component Analysis*. Essentially for many applications, the proposed method makes PCA tracatable on large datasets. We defer these derivations to future work.

Borrowing ideas from kernel approximation, we can approximate the labels kernel $L = \Psi^T\Psi$, which results in an updated form for Q

$$Q = XHLHX^T \tag{5.6}$$
$$= \underbrace{XH\Psi^T}_{V\Sigma^{\frac{1}{2}}}\underbrace{\Psi HX^T}_{\Sigma^{\frac{1}{2}}V^T} \tag{5.7}$$

where V and Σ contain the eigenvectors of eigenvalues of Q respectively as obtained from SVD where $Q = V\Sigma V^T$. Recall also from Section 5.2.2 that $V = U$ is the optimal solution of the optimization problem for SPCA (i.e., (5.5)). Therefore, we have that $U\Sigma^{\frac{1}{2}}$ and $\hat{U} = XH\Psi^T$ are equal up to a rotation. This means that $\hat{U}$ is not identical to U, but is a projection whose columns are scaled according to the square-root of the singular values of Q. This method will henceforth be referred to as *Supervised Random Projections* (SRP). Recall that in SPCA, when the data was to be down-projected into k dimensions, the eigenvectors corresponding to the top k eigenvalues were selected in U. In SRP, however, we have

$$\hat{U}_{d\times k} = X_{d\times n}\ H_{n\times n}\ \Psi^T_{n\times k} \tag{5.8}$$

which suggests that using k random bases in Ψ when approximating the L kernel will down-project the data into k dimensions, as follows:

[2] In such a case, the L kernel is set equal to the identity matrix, i.e., a kernel which only captures the similarity between a point and itself. Therefore, (from [4]), Q becomes the covariance matrix of mean-subtracted samples X, and decomposing the covariance matrix is the same as decomposing Q and consequently the same as maximizing $\mathbf{tr}(U^TQU)$. In other words, setting $L = I$ means that we retain the maximal diversity between observations, and therefore PCA is a special case of SCPA.

$$\begin{aligned} Q &= XHLHX^T \\ &= XHIHX^T \\ &= (XH)(XH)^T \\ &= (X-\mu_x)(X-\mu_x)^T \\ &= Cov(X) \end{aligned}$$

$$\hat{X} = \hat{U}^T X \tag{5.9}$$
$$= \Psi H X^T X \tag{5.10}$$

where $\hat{X}$ is the down-projected data. A simple extension of the above linear down-projection to replace $X^T X$ with a kernelized version of the data, i.e.,

$$\hat{X} = \Psi H K \tag{5.11}$$

which shall be referred to as *Kernel Supervised Random Projections* (KSRP).

5.3 Experiments

5.3.1 Datasets

The motivation of this chapter is to make tractable the use of Supervised Principal Component Analysis (SPCA) and Kernel Supervised Principal Component Analysis (KSPCA) for large-scale data applications, hence the introduction of Supervised Random Projections (SRP) and Kernel Supervised Random Projections (KSRP). Therefore, it is pertinent to experiment with a variety of datasets: small datasets, so to compare the proposed method with the baseline; large datasets, which become intractable for SPCA; datasets with obvious nonlinearities, so to compare the performance of linear methods (i.e., SPCA and SRP) with nonlinear methods (i.e., KSPCA and KSRP). For this purpose, we experiment with the following datasets: i) UCI-Ionosphere[3] [69] as was used in [4], ii) MNIST[4] [48], and iii) synthetic XOR dataset[5].

[3] 251 34-dimensional samples; 2-class; %70/%30 train/test split.

[4] $12,500$ 784-dimensional samples; 10-class; %80/%20 train/test split. Although MNIST contains $60,000$ training samples and $10,000$ test samples, we randomly sampled $1,000$ train and 250 test samples from each class to form our dataset. This was done to allow for tractable computation of the U in SPCA.

[5] 500 10-dimensional samples; 2-class; %70/%30 train/test split. The synthetic 2-class XOR dataset was constructed by generating 500 2-dimensional multivariate normal random variables with identity covarince and four different means at $\{(\alpha,\alpha),(-\alpha,\alpha),(\alpha,-\alpha),(-\alpha,-\alpha)\}$ where $\alpha > 1$ to prevent blending, in our case was set to 5. The samples generated from the distributions with the first and third mean were set to class 1, and the other samples were set to class 2. Finally, 8-dimensional random noise was added independently to each sample, yielding the final 500 10-dimensional samples.

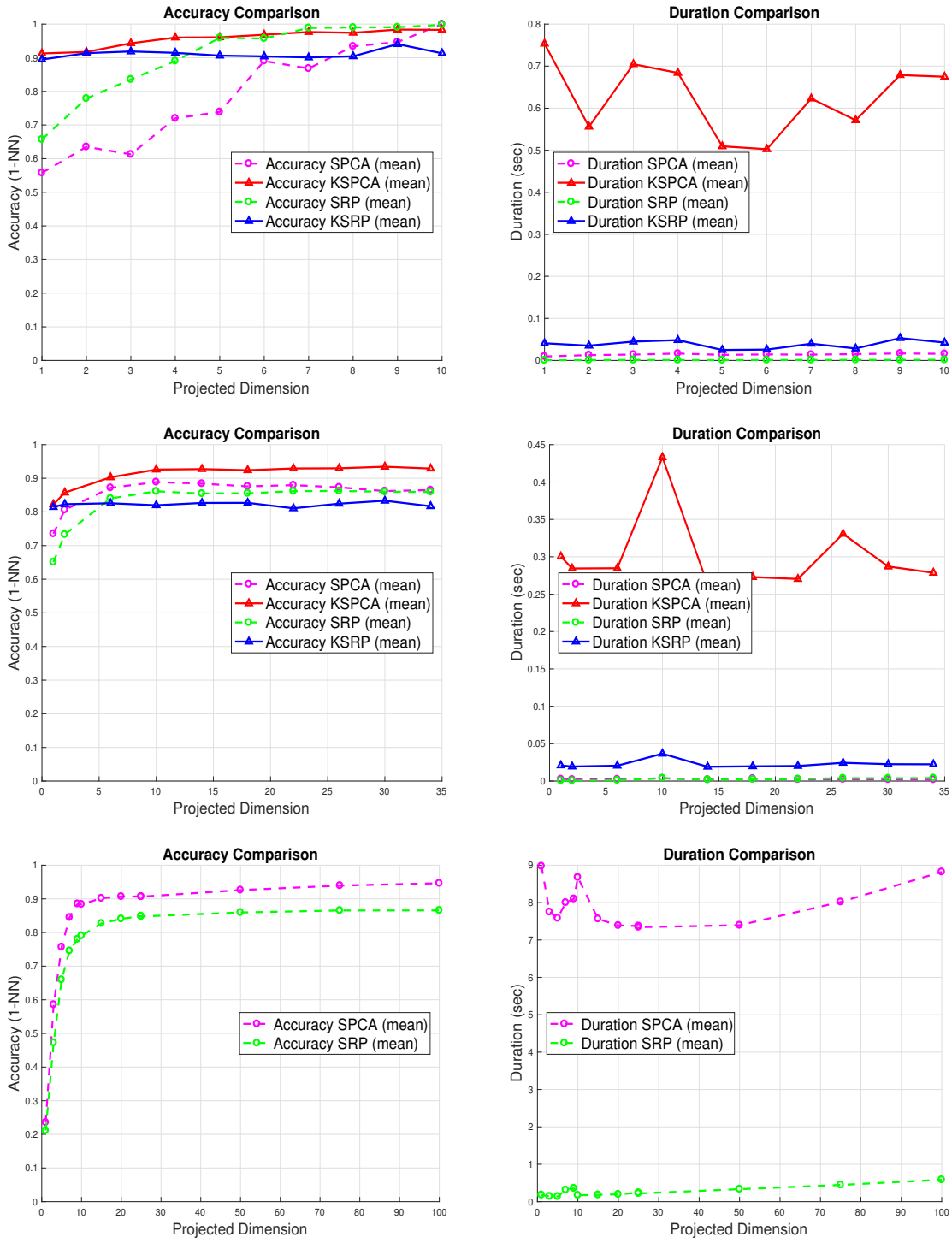


Figure 5.1: 1-NN and time performance comparison on synthetic XOR (top-row), UCI-Ionosphere (middle row), and MNIST (bottom row) datasets. The left figures show 1-NN performance whereas the right plots compare time performance. All results are averaged over 40 runs.

5.3.2 Metrics

As with previous chapters, because we are assessing the quality of the embedding in the setting of dimensionality reduction, we shall employ 1-Nearest Neighbor (1-NN) classification to measure and compare the performance of the embeddings obtained from each method. Furthermore, we keep track of the clock-time duration of each dimensionality reduction setup and compare for efficiency. To measure the clock-time, we only consider the duration for computing the projection U, and do not include the time for finding the embedding or the time required for computing 1-NN classification performance, as these are highly dependant on the dimensionality of the embedding space.

5.3.3 Setup

The results presented here compare the 1-NN and time performances of SPCA, SRP, KSPCA, and KSRP on the aforementioned datasets. The results are averaged over 40 independent runs of the same script, and the datasets were either generated anew in each run (in the case of synthetic XOR), or different slices of the dataset were randomly selected for train/test from the available data (in the case of UCI-Ionosphere and MNIST). Each feature of all data samples was mean-subtracted based on the mean of the features in the training samples. Figure 5.1 shows the results for the synthetic XOR, UCI-Ionosphere, and subsampled MNIST datasets. Confidence bounds were measured over 40 trials, but not shown on the plots for clarity (plots become messy due to overlapping regions), however, they are all comparatively small.

5.4 Discussion

When analyzing the results of Figure 5.1, we make a number of interesting observations. Firstly, we note that increasing the number of random bases / projected dimension (i.e., k) results in better 1-NN performance. This is expected because with higher ks, we are retaining more information about the dataset in the embedding space. Assuming the test data is sampled from the same distribution as the training data (which may include the same noise distribution), higher k should result in better performance 1-NN on the test set in the embedding space.

When comparing the time performances of various methods, we immediately notice the burdensome compute time required for KSPCA compared to that of KSRP and linear SPCA and SRP. We should keep in mind that, as mentioned in Section 5.2.1, there are kernel approximation schemes that are much faster than Random Kitchen Sinks which was

Table 5.1: Comparing time complexities of SPCA, KSPCA, SRP, and KSRP. Reminder: $X_{d\times n}$, $K_{n\times n}$, $L_{n\times n}$, $H_{n\times n}$, $\Psi_{k\times n}$ where n is the number of training samples, d is the original dimensionality, and k is the embedding space dimensionality or the number of random bases. The subscript under Ψ refers to whether we are approximating the data kernel K (via explicit mapping Ψ_X with k_x random bases) or the labels kernel L (via explicit mapping Ψ_Y with k_y random bases).

Method	Matrix Multiplication	Kernel Computation (K, Ψ)	SVD
$U_{SPCA} = eig(XHLHX^T)$	$\mathcal{O}(\max\{d^2n, dn^2\})$	-	$\mathcal{O}(d^3)$
$U_{KSPCA} = eig(KHLHK^T)$	$\mathcal{O}(n^3)$	$\mathcal{O}(dn^2)$	$\mathcal{O}(n^3)$
$\hat{U}_{SRP} = XH\Psi_Y^T$	$\mathcal{O}(\max\{dn^2, k_yn^2\})$	$\mathcal{O}(ndk_y)$	-
$\hat{U}_{KSRP} = \Psi_X H\Psi_Y^T$	$\mathcal{O}(\max\{k_xn^2, k_yn^2\})$	$\mathcal{O}(\max\{nkd_x, nkd_y\})$	-

used in our setup. This suggests that the efficiency gains observed here can be even more dramatic if we employ a kernel approximation method such as FastFood [47]. Furthermore, it is interesting to observe that as the projected dimension k increases, the time required for SPCA and KSPCA neither increases or decreases (as expected, because the SVD remains unchanged) while the time complexity of SRP and KSRP increases as the number of random bases increase (e.g., bottom-right plot of Figure 5.1).

It is worth reiterating that the SVD stage of SPCA scales with $\mathcal{O}(d^3)$ (i.e., the dimensionality of the data) whereas KSPCA's SVD stage scales with $\mathcal{O}(n^3)$ (i.e., the cardinality of the dataset). This is because, in SPCA, $Q = XHLHX^T$ is decomposed, whereas in KSPCA $Q = KHLHK^T$ is decomposed using SVD [4]. This explains why KSPCA should perform worse than SPCA in time complexity: i) the datasets that we experiment with typically contain many more training samples than features, and ii) KSPCA must construct the data kernel matrix K in addition to performing SVD. This further elucidates the competitive advantage of using KSRP which is both nonlinear compared to SPCA, and has faster run-time than KSPCA as it bypasses kernel construction altogether. In fact, Table 5.1 compares the complexities of these method [6]. Bearing in mind that the number of samples of many datasets is typically larger than the dimensionality of the samples (i.e., $n > d$), this shows that while KSPCA has a total complexity of $\mathcal{O}(n^3)$, SPCA and SRP have a complexity of $\mathcal{O}(dn^2)$ and KSRP has a time complexity of $\mathcal{O}(\max\{k_xn^2, k_yn^2\})$. This aligns perfectly with the observed time duration results of Figure 5.1.

For the synthetic XOR plots (i.e., bottom row) of Figure 5.1, it agrees with our intuition that KSPCA and KSRP outperform their linear counterparts in lower dimensions. For

[6] Multiplying an $m \times n$ matrix by an $n \times p$ matrix has complexity $\mathcal{O}(mnp)$.

the subsampled MNIST plots (i.e., bottom row) of Figure 5.1, we omitted the results of KSPCA as they were very expensive to compute while they did not give the best performing embedding: initial experiments suggested KSPCA required almost $100\times$ more time to run, while the performance was worse than SPCA. For this latter reason, we also do not report the performance of KSRP. These plots do demonstrate the superior performance of SPCA compared ro SRP, but at the cost of $40\times$ increased time complexity.

5.5 Conclusion

Previous chapters saw various data-independent random projection schemes for dimensionality reduction. In contrast, this chapter suggests the use of label information present in some datasets to assist in the random projection by finding an embedding that maximizes the dependence of the labels on the embedding data. To achieve this, we are inspired by Supervised Principal Component Analysis (SPCA) and Kernel Supervised Principal Component Analysis (KSPCA) where an optimization function is solved in closed-form where the optimal solution is the desired embedding matrix U. This chapter builds on the theory of SPCA and that of kernel approximation to construct Supervised Random Projections (SRP) and Kernel Supervised Random Projections (KSRP). Evaluated on 3 different datasets for classification, SRP and KSRP performed very competitively with SPCA and KSPCA, yielding a small drop in 1-NN classification performance while providing orders of magnitude better time performance.

Chapter 6

Conclusion

In this chapter, a brief summary of the thesis and the key contributions are described in Section 6.1. Potential future work for this research is outlined in Section 6.2.

6.1 Summary of Thesis and Contributions

The general motivation of this thesis concerned the design of new forms of random projection for prediction and dimensionality reduction in big-data domains where traditional methods suffer from exponentially more expensive compute power required. Such domains include classification of high-dimensional but scarce data samples, and dimensionality reduction of high-dimensional natural data. In this regard, this thesis explored new forms of linear and nonlinear random projection setups and comprehensively tested and compared these setups against traditional methods.

In Chapter 3, we explored an alternative method for linear random projections that were inspired by distributions of biological synapses observed in the human visual cortex and studied its application for classifying high-dimensional datasets with few samples. Here we constructed linear random projections with the goal of performing prediction (classification) on natural imaging data. With this in mind, the coefficients of the random matrices used for projection were spatially correlated to extract and make use of local information in the original images. We constructed two new linear random projection setups and compared their classification performance using a standard multi-layered perceptron classifier in Table 3.1. The results presented a powerful first step towards designing deep neural networks that do not require many data samples to learn, and can sidestep / reduce

the burden of current training procedures while maintaining or boosting classification and modelling performance.

In Chapter 4, we extended the ideas of Chapter 3 and proposed an extension of linear random projections called *ensemble of nonlinear maximum random projections*. Inspired by ideas in the literature on nonlinear random projections [25], we showed that this form of nonlinear random projection is well-suited for dimensionality reduction of highly complex natural datasets. Based on the theory in [25], we argued that the resulting embedding differs from traditional linear random projection embeddings in that pairwise distances are not necessarily preserved for all pairs of points. Instead, based on the common assumption that data samples from the same class have smaller angular distance with higher probability, the proposed form of nonlinear random projection creates an embedding where inter-class distance while reducing intra class distance, essentially individually clustering each sample with members of its own class making it ideal for downstream applications such as classificaiton. Table 4.1 showed a comparison of the proposed method against traditional linear random projections, nonlinear rectified random projections, and principal component analysis, detailing the superior classification performance of the proposed nonlinear method for high-dimensionsal data.

Finally, in Chapter 5.2.3 introduces the use of label information present in some datasets to assist in the random projection by finding an embedding that maximizes the dependence of the labels on the embedding data. To achieve this, we are inspired by Supervised Principal Component Analysis (SPCA) and Kernel Supervised Principal Component Analysis (KSPCA) where an optimization function is solved in closed-form where the optimal solution is the desired embedding matrix U. This chapter builds on the theory of SPCA and that of kernel approximation to construct Supervised Random Projections (SRP) and Kernel Supervised Random Projections (KSRP). Evaluated on 3 different datasets for classification, SRP and KSRP performed very competitively with SPCA and KSPCA, yielding a small drop in 1-NN classification performance while providing orders of magnitude better time performance (Figure 5.1).

6.2 Future Work

6.2.1 Random Projections for Larger Networks and Data Regimes

There are a number of exciting avenues of future research that build on random projections from biologically inspired distributions. Firstly, we are excited to explore this same effect on deeper networks with more synapses, and to investigate how and whether these synaptic

strength distributions may be used to design more efficient architectures and training algorithms. Another interesting direction is to explore the effect of increasing datasets size on classification performances — we expect that the performance of random-weighted neural networks to break down as we move to operate in larger datasets regimes. Finally, the current results were inspired by synapse distributions emperically observed in the visual cortex of humans, however, it is important that future research ground these observations in theory of random projections (similar to Appendix A) as such theory may suggest newer forms of random projections and derivations thereof that have not yet been explored.

6.2.2 Nonlinear Random Projections Theory

In future work, we would like to derive a theory for the probability bounds of the nonlinear embedding of samples into lower dimensions using the max activation function. Specifically, we would like to assert the claim of [25] that this bound is similar for spatial pooling and ReLU activation functions, and to explore the differences and the interplay between these two nonlinearities, and variants thereof. Additionally, our preliminary experiments on multi-layered nonlinear RPs (not included here for brevity) hint at the compounded power of such projections in further boosting performance. Theory in this vein is promising for theoretically backing empirical results observed by [38] and [42] where CNNs with random weights competitively performed on classification tasks.

6.2.3 Random Proxy Networks

An exciting direction of research is using random-weighted neural networks as a proxy for model selection. To evaluate a Deep Neural Network (DNN), it must first be trained, and then the generalizability of the model is assessed on unseen data. DNN training is iterative and can take many hours and consume many resources, especially as models become more complex. In my thesis work, I have demonstrated the effectiveness of random weighted networks in resource-constrained scenarios, such as embedded hardware. These networks are DNNs with any desired architecture, with random untrained weights, positioning them as very efficient alternatives to trained DNNs. Therefore, exploring the application of such networks as a proxy for evaluating the performance of a DNN seems like a natural next step, and is actively being explore by members of the community [29, 68].

6.2.4 Convolutional Random Projections

Finally, in addition to multi-layered random projections, it is pertinent to study convolutional random projections (i.e., random weights used in convolutional kernels). In Chapter 3, we studied a setup where coefficients of *convolutional* kernels were sampled from biological distributions, because convolutional architectures are also loosely inspired by the visual cortex. This played well with the nature of the imaging datasets that were experimented. Further studies can dive deep into the various parameters that effect the performance of convolutional kernels such as width, stride, number of kernels, etc.

References

[1] Dimitris Achlioptas. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *Journal of computer and System Sciences*, 66(4):671–687, 2003.

[2] Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, pages 459–468. IEEE, 2006.

[3] Boris Barbour, Nicolas Brunel, Vincent Hakim, and Jean-Pierre Nadal. What can we learn from synaptic weight distributions? *TRENDS in Neurosciences*, 30(12):622–629, 2007.

[4] Elnaz Barshan, Ali Ghodsi, Zohreh Azimifar, and Mansoor Zolghadri Jahromi. Supervised principal component analysis: Visualization, classification and regression on subspaces and submanifolds. *Pattern Recognition*, 44(7):1357–1371, 2011.

[5] Emmanuel J Candes. The restricted isometry property and its implications for compressed sensing. *Comptes rendus mathematique*, 346(9-10):589–592, 2008.

[6] Emmanuel J Candes and Terence Tao. Decoding by linear programming. *IEEE transactions on information theory*, 51(12):4203–4215, 2005.

[7] Emmanuel J Candes and Terence Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE transactions on information theory*, 52(12):5406–5425, 2006.

[8] Soon Keen Cheong, Chris Tailby, Samuel G Solomon, and Paul R Martin. Cortical-like receptive fields in the lateral geniculate nucleus of marmoset monkeys. *Journal of Neuroscience*, 33(16):6864–6876, 2013.

[9] Audrey G Chung, Mohammad Javad Shafiee, and Alexander Wong. Random feature maps via a layered random projection (larp) framework for object classification. In

Image Processing (ICIP), 2016 IEEE International Conference on, pages 246–250. IEEE, 2016.

[10] Adam Coates, Andrew Y Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *International conference on artificial intelligence and statistics*, pages 215–223, 2011.

[11] Scott Cost and Steven Salzberg. A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning*, 10:57–67, 1993.

[12] T.M. Cover and Peter E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, pages 21–27, 1967.

[13] David Cox and Nicolas Pinto. Beyond simple features: A large-scale feature search approach to unconstrained face recognition. In *Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 8–15. IEEE, 2011.

[14] Sanjoy Dasgupta. Experiments with random projection. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pages 143–151. Morgan Kaufmann Publishers Inc., 2000.

[15] Sanjoy Dasgupta and Anupam Gupta. An elementary proof of a theorem of johnson and lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65, 2003.

[16] Jason V Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S Dhillon. Information-theoretic metric learning. In *Proceedings of the 24th international conference on Machine learning*, pages 209–216. ACM, 2007.

[17] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the Association for Information Sciences and Technology*, 41:391–407, 1990.

[18] David L Donoho. Compressed sensing. *IEEE Transactions on information theory*, 52(4):1289–1306, 2006.

[19] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification and Scene Analysis*. John Wiley and Sons, 1973.

[20] Chris Eliasmith, Terrence C Stewart, Xuan Choo, Trevor Bekolay, Travis DeWolf, Yichuan Tang, and Daniel Rasmussen. A large-scale model of the functioning brain. *science*, 338(6111):1202–1205, 2012.

[21] Christos Faloutsos and Douglas W. Oard. A survey of information retrieval and filtering methods. *Technical Report*, 1995.

[22] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer vision and Image understanding*, 106(1):59–70, 2007.

[23] Peter Frankl and Hiroshi Maehara. The johnson-lindenstrauss lemma and the sphericity of some graphs. *Journal of Combinatorial Theory, Series B*, 44(3):355–362, 1988.

[24] Allen Gersho, Robert M. Gray, and David G. Stork. *Vector quantization and signal compression*. Kluwer, 1991.

[25] Raja Giryes, Guillermo Sapiro, and Alex M Bronstein. Deep neural networks with random gaussian weights: A universal classification strategy. 2015.

[26] Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In *International conference on algorithmic learning theory*, pages 63–77. Springer, 2005.

[27] Raffay Hamid, Ying Xiao, Alex Gittens, and Dennis DeCoste. Compact random feature maps. In *International Conference on Machine Learning*, pages 19–27, 2014.

[28] Trevor Hastie and Robert Tibshirani. Discriminant adaptive nearest neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18, 1996.

[29] Kun He, Yan Wang, and John Hopcroft. A powerful generative model using random weights for the deep image representation. In *Advances in Neural Information Processing Systems*, pages 631–639, 2016.

[30] Michael Holmes and Alexander Gray. Fast svd for large-scale matrices.

[31] Guang-Bin Huang, Lei Chen, Chee Kheong Siew, et al. Universal approximation using incremental constructive feedforward networks with random hidden nodes. *IEEE Trans. Neural Networks*, 17(4):879–892, 2006.

[32] Guang-Bin Huang, Dian Hui Wang, and Yuan Lan. Extreme learning machines: a survey. *International journal of machine learning and cybernetics*, 2(2):107–122, 2011.

[33] Guang-Bin Huang, Hongming Zhou, Xiaojian Ding, and Rui Zhang. Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 42(2):513–529, 2012.

[34] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: theory and applications. *Neurocomputing*, 70(1-3):489–501, 2006.

[35] David H Hubel and Torsten N Wiesel. Receptive fields and functional architecture of monkey striate cortex. *The Journal of physiology*, 195(1):215–243, 1968.

[36] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613. ACM, 1998.

[37] BA J Mercer. Xvi. functions of positive and negative type, and their connection the theory of integral equations. *Phil. Trans. R. Soc. Lond. A*, 209(441-458):415–446, 1909.

[38] Kevin Jarrett, Koray Kavukcuoglu, Yann LeCun, et al. What is the best multi-stage architecture for object recognition? In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2146–2153. IEEE, 2009.

[39] William B Johnson and Joram Lindenstrauss. Extensions of lipschitz mappings into a hilbert space. *Contemporary mathematics*, 26(189-206):1, 1984.

[40] Sham Kakade and Greg Shakhnarovich. Random projections (cmsc 35900 large scale learning), 2009.

[41] Purushottam Kar and Harish Karnick. Random feature maps for dot product kernels. In *Artificial Intelligence and Statistics*, pages 583–591, 2012.

[42] Amir-Hossein Karimi, MJ Shafiee, A Ghodsi, and A Wong. Synthesizing deep neural network architectures using biological synaptic strength distributions. *Computational Cognitive Neuroscience (CCN)*, 2017.

[43] Liyanaarachchi Lekamalage Chamara Kasun, Yan Yang, Guang-Bin Huang, and Zhengyou Zhang. Dimension reduction with extreme learning machine. *IEEE Transactions on Image Processing*, 25(8):3906–3918, 2016.

[44] Liyanaarachchi Lekamalage Chamara Kasun, Hongming Zhou, Guang-Bin Huang, and Chi Man Vong. Representational learning with elms for big data. 2013.

[45] Kyungnam Kim. Face recognition using principal component analysis. In *International Conference on Computer Vision and Pattern Recognition*, pages 586–591, 1996.

[46] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images, 2009.

[47] Quoc Le, Tamás Sarlós, and Alex Smola. Fastfood-approximating kernel expansions in loglinear time.

[48] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[49] Ping Li, Trevor J Hastie, and Kenneth W Church. Very sparse random projections. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 287–296. ACM, 2006.

[50] Ik Soo Lim, Pablo de Heras Ciechomski, Sofiane Sarni, and Daniel Thalmann. Planar arrangement of high-dimensional biomedical data sets by isomap coordinates. In *Computer-Based Medical Systems, 2003. Proceedings. 16th IEEE Symposium*, pages 50–55. IEEE, 2003.

[51] Amaro Lima, Heiga Zen, Yoshihiko Nankaku, Chiyomi Miyajima, Keiichi Tokuda, and Tadashi Kitamura. On the use of kernel pca for feature extraction in speech recognition. *IEICE TRANSACTIONS on Information and Systems*, 87(12):2802–2811, 2004.

[52] Li Liu and Paul Fieguth. Texture classification from random features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(3):574–586, 2012.

[53] Michael Mahoney, Ben Newhouse, and Gourab Mukherjee. The johnson-lindenstrauss lemma (cs369m: Algorithms for modern massive data set analysis), 2009.

[54] Stéphane Mallat. *A wavelet tour of signal processing*. Academic press, 1999.

[55] Luis M Martinez and Jose-Manuel Alonso. Complex receptive fields in primary visual cortex. *The neuroscientist*, 9(5):317–331, 2003.

[56] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 5. Granada, Spain, 2011.

[57] Ninh Pham and Rasmus Pagh. Fast and scalable polynomial kernels via explicit feature maps. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 239–247. ACM, 2013.

[58] Nicolas Pinto, David Doukhan, James J DiCarlo, and David D Cox. A high-throughput screening approach to discovering good forms of biologically inspired visual representation. *PLoS Comput Biol*, 5(11):e1000579, 2009.

[59] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

[60] Bisser Raytchev, Ikushi Yoda, and Katsuhiko Sakaue. Head pose estimation by non-linear manifold learning. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 4, pages 462–466. IEEE, 2004.

[61] Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.

[62] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 2015.

[63] Gerard Salton and Michael J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill Inc., 1983.

[64] Guido Sanguinetti. Dimensionality reduction of clustered data sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(3):535–540, 2008.

[65] Andrew Saxe, Pang W Koh, Zhenghao Chen, Maneesh Bhand, Bipin Suresh, and Andrew Y Ng. On random weights and unsupervised feature learning. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 1089–1096, 2011.

[66] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Kernel principal component analysis. In *International Conference on Artificial Neural Networks*, pages 583–588. Springer, 1997.

[67] Bernhard Scholkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

[68] Yao Shu, Man Zhu, Kun He, John Hopcroft, and Pan Zhou. Understanding deep representations through random weights. *arXiv preprint arXiv:1704.00330*, 2017.

[69] Vincent G Sigillito, Simon P Wing, Larrie V Hutton, and Kile B Baker. Classification of radar returns from the ionosphere using neural networks. *Johns Hopkins APL Technical Digest*, 10(3):262–266, 1989.

[70] Aman Sinha and John C Duchi. Learning kernels with random features. In *Advances in Neural Information Processing Systems*, pages 1298–1306, 2016.

[71] Sen Song, Per Jesper Sjöström, Markus Reigl, Sacha Nelson, and Dmitri B Chklovskii. Highly nonrandom features of synaptic connectivity in local cortical circuits. *PLoS Biol*, 3(3):e68, 2005.

[72] Jiexiong Tang, Chenwei Deng, and Guang-Bin Huang. Extreme learning machine for multilayer perceptron. *IEEE transactions on neural networks and learning systems*, 27(4):809–821, 2016.

[73] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

[74] Warren S Torgerson. Multidimensional scaling: I. theory and method. *Psychometrika*, 17(4):401–419, 1952.

[75] Laurens Van Der Maaten, Eric Postma, and Jaap Van den Herik. Dimensionality reduction: a comparative. *J Mach Learn Res*, 10:66–71, 2009.

[76] Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb):207–244, 2009.

[77] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

[78] Rui Xu, Steven Damelin, and Donald C Wunsch. Applications of diffusion maps in gene expression data-based cancer diagnosis analysis. In *Engineering in medicine and biology society, 2007. EMBS 2007. 29th annual international conference of the IEEE*, pages 4613–4616. IEEE, 2007.

[79] Shuicheng Yan, Dong Xu, Benyu Zhang, Hong-Jiang Zhang, Qiang Yang, and Stephen Lin. Graph embedding and extensions: A general framework for dimensionality reduction. *IEEE transactions on pattern analysis and machine intelligence*, 29(1):40–51, 2007.

[80] Ji Zhu and Trevor Hastie. Kernel logistic regression and the import vector machine. *Journal of Computational and Graphical Statistics*, 14(1):185–205, 2005.

APPENDICES

Appendix A

Proof of JL Lemma for Gaussian Random Matrices

Appendix A details the proof of the Johnson-Lindenstrauss lemma for Gaussian random matrices. This section is inspired by the many great work done by [39, 15, 1, 40, 53]. As a reminder,

Theorem 2. *For any point-set* $\mathcal{P} = \{\boldsymbol{x_1}, \boldsymbol{x_2}, \cdots, \boldsymbol{x_n}\} \subset \mathbb{R}^d$*, any integer* n *(number of samples), and any* $0 < \epsilon < 1$ *(error tolerance), let* k *be a positive integer satisfying*

$$k \geq 4(\epsilon^2/2 - \epsilon^3/3)^{-1} \log n \tag{A.1}$$

then, there exists a map $f : \mathbb{R}^d \to \mathbb{R}^k$ *such that for all* $\boldsymbol{x_i}, \boldsymbol{x_j} \in \mathcal{P}$*, with probability greater than* $1 - \delta$ *we have*

$$(1-\epsilon)||\boldsymbol{x_i} - \boldsymbol{x_j}||^2 \leq ||f(\boldsymbol{x_i}) - f(\boldsymbol{x_j})||^2 \leq (1+\epsilon)||\boldsymbol{x_i} - \boldsymbol{x_j}||^2 \tag{A.2}$$

Here, we walk through a simple example for Gaussian random projections where all coefficients of the projection matrix $R \in \mathbb{R}^{d \times k}$ are sampled independently from a standard normal distribution, i.e., $r_{ij} \sim \mathcal{N}(0, 1/k)$ or equivalently $r_{ij} \sim \frac{1}{\sqrt{k}}\mathcal{N}(0, 1)$. The proof follows the steps below:

1. Show for fixed $\boldsymbol{x_1}$, $\boldsymbol{x_2}$: $\mathbb{E}[||f(\boldsymbol{x_1}) - f(\boldsymbol{x_2})||_2^2] = ||\boldsymbol{x_1} - \boldsymbol{x_2}||_2^2$
2. Show variance is bounded and small for $\boldsymbol{x_1}$, $\boldsymbol{x_2}$
3. Using Bonferroni's Union Bound, bound the failure probability for all pairs of points

A.1 Step 1. Show $\mathbb{E}[||f(\boldsymbol{x_1}) - f(\boldsymbol{x_2})||_2^2] = ||\boldsymbol{x_1} - \boldsymbol{x_2}||_2^2$

$$
\begin{aligned}
\mathbb{E}\Big[||f(\boldsymbol{x_1}) - f(\boldsymbol{x_2})||_2^2\Big] &= \mathbb{E}\Big[||R^T\boldsymbol{x_1} - R^T\boldsymbol{x_2}||_2^2\Big] \\
&= \mathbb{E}\Big[\sqrt{\sum_{l=1}^{k}\Big((R_l^T\boldsymbol{x_1} - R_l^T\boldsymbol{x_2})^2\Big)}^{\,2}\Big] \\
&= \mathbb{E}\Big[\sum_{l=1}^{k}\Big((R_l^T\boldsymbol{x_1} - R_l^T\boldsymbol{x_2})^2\Big)\Big] \\
&= \sum_{l=1}^{k}\Big(\mathbb{E}\Big[(R_l^T\boldsymbol{x_1} - R_l^T\boldsymbol{x_2})^2\Big]\Big) \\
&= \sum_{l=1}^{k}\Big(\underbrace{\mathbb{E}\Big[(R_l^T\boldsymbol{x_1})^2\Big]}_{\mathbf{A}} + \underbrace{\mathbb{E}\Big[(R_l^T\boldsymbol{x_2})^2\Big]}_{\mathbf{B}} - 2\underbrace{\mathbb{E}\Big[R_l^T\boldsymbol{x_1}\ R_l^T\boldsymbol{x_2}\Big]}_{\mathbf{C}}\Big)
\end{aligned}
$$

Deriving A:[a]

$$
\begin{aligned}
\mathbb{E}\Big[(R_l^T\boldsymbol{x_1})^2\Big] &= \mathbb{E}\Big[(\sum_{i=1}^{d} r_{li}x_{1i})^2\Big] \\
&= \mathbb{E}\Big[\sum_{i=1}^{d}\sum_{j=1}^{d} r_{li}r_{lj}x_{1i}x_{1j}\Big] \\
&= \sum_{i=1}^{d}\sum_{j=1}^{d}\mathbb{E}\Big[r_{li}r_{lj}x_{1i}x_{1j}\Big] \\
&= \sum_{i=1}^{d}\sum_{j=1}^{d} x_{1i}x_{1j}\mathbb{E}\Big[r_{li}r_{lj}\Big] \\
&= \sum_{i=1}^{d}\sum_{j=1}^{d} x_{1i}x_{1j}\delta_{ij} \\
&= \frac{1}{k}\sum_{i=1}^{d} x_{1i}^2 \\
&= \frac{||\boldsymbol{x_1}||_2^2}{k}
\end{aligned}
$$

Deriving B:

$$
\begin{aligned}
\mathbb{E}\Big[(R_l^T\boldsymbol{x_2})^2\Big] &= \mathbb{E}\Big[(\sum_{i=1}^{d} r_{li}x_{1i})^2\Big] \\
&= \cdots \\
&= \frac{1}{k}\sum_{i=1}^{d} x_{2i}^2 \\
&= \frac{||\boldsymbol{x_2}||_2^2}{k}
\end{aligned}
$$

Deriving C:

$$
\begin{aligned}
\mathbb{E}\Big[R_l^T\boldsymbol{x_1}\ R_l^T\boldsymbol{x_2}\Big] &= \mathbb{E}\Big[(\sum_{i=1}^{d} r_{li}x_{1i})\cdot(\sum_{j=1}^{d} r_{lj}x_{2j})\Big] \\
&= \cdots \\
&= \frac{1}{k}\sum_{i=1}^{d} x_{1i}x_{2i} \\
&= \frac{\langle\boldsymbol{x_1}, \boldsymbol{x_2}\rangle}{k}
\end{aligned}
$$

[a] $\delta_{ij} = \begin{cases} 0, & i \neq j \\ 1/k, & i = j \end{cases}$

Putting it all together, we get:

$$\begin{aligned}
\implies \mathbb{E}\Big[||f(\boldsymbol{x_1}) - f(\boldsymbol{x_2})||_2^2\Big] &= \sum_{l=1}^{k} \Big(\mathbf{A} + \mathbf{B} - 2\mathbf{C}\Big) \\
&= \sum_{l=1}^{k} \left(\frac{||\boldsymbol{x_1}||_2^2}{k} + \frac{||\boldsymbol{x_2}||_2^2}{k} - 2\frac{\langle \boldsymbol{x_1}, \boldsymbol{x_2} \rangle}{k}\right) \\
&= \sum_{l=1}^{k} \left(\frac{||\boldsymbol{x_1} - \boldsymbol{x_2}||_2^2}{k}\right) \\
&= ||\boldsymbol{x_1} - \boldsymbol{x_2}||_2^2 \qquad \square
\end{aligned}$$

Note that all we require for this step of the proof is independence and unit variance in constructing the random matrix R, suggesting that there are other distribution families other than standard normal that support random projections.

A.2 Step 2. Show variance is bounded and small for one pair of points

In this step, we attempt to bound the probability of failure. Hence, we start with:

$$\begin{aligned}
Pr\Big[||f(\boldsymbol{x_1}) - f(\boldsymbol{x_2})||^2 > (1+\epsilon)||\boldsymbol{x_1} - \boldsymbol{x_2}||^2\Big] &= Pr\Big[||R^T\boldsymbol{x_1} - R^T\boldsymbol{x_2}||^2 > (1+\epsilon)||\boldsymbol{x_1} - \boldsymbol{x_2}||^2\Big] \\
&= Pr\Big[||R^T(\boldsymbol{x_1} - \boldsymbol{x_2})||^2 > (1+\epsilon)||\boldsymbol{x_1} - \boldsymbol{x_2}||^2\Big] \\
&= Pr\Big[||R^T\boldsymbol{x_d}||^2 > (1+\epsilon)||\boldsymbol{x_d}||^2\Big]
\end{aligned}$$

where we observe that the random projection matrix R is doing a linear operation, and so the second line follows from the distributive property of linear matrix-vector multiplication. Let's now define $\boldsymbol{y} = R^T\boldsymbol{x}$ as the embedding vector for $\boldsymbol{x}$, and recall that R is a $k \times d$ matrix, where each entry is sampled i.i.d from a Gaussian $\mathcal{N}(0, 1/k)$. Furthermore, note

that each coefficient of $\boldsymbol{y}$ ($y_l = R_{l.}^T \boldsymbol{x}\ \forall\ i \in [k]$) is distributed as $\frac{1}{\sqrt{k}}\mathcal{N}(0, ||\boldsymbol{x}||^2)$, and y_l are independent. Alternatively, we can define A as a $k \times d$ matrix, where each entry is sampled i.i.d from a Gaussian $\mathcal{N}(0,1)$ and conclude that each coefficient of $\boldsymbol{y}$ ($y_l = \frac{1}{\sqrt{k}} A_{l.}^T \boldsymbol{x}\ \forall\ i \in [k]$) is distributed as $\frac{1}{\sqrt{k}}\mathcal{N}(0, ||\boldsymbol{x}||^2)$ or that $z_l = A_{l.}^T \boldsymbol{x}/||\boldsymbol{x}||$ is distributed as $\mathcal{N}(0,1)$. Hence we have:

$$
\begin{aligned}
Pr\Big[||R^T\boldsymbol{x}||^2 > (1+\epsilon)||\boldsymbol{x}||^2\Big] &= Pr\Big[||\frac{1}{\sqrt{k}}A^T\boldsymbol{x}||^2 > (1+\epsilon)||\boldsymbol{x}||^2\Big] \\
&= Pr\Big[||\boldsymbol{z}||^2 > (1+\epsilon)k\Big] \\
&= Pr\Big[\sum_{l=1}^{k} z_l^2 > (1+\epsilon)k\Big] \\
&= Pr\Big[\chi_k^2 > (1+\epsilon)k\Big]
\end{aligned}
$$

where χ_k^2 is the chi-squared distribution with k degrees of freedom. To solve this, we make use of the Markov Inequality[1] and the moment generating function of the chi-squared distribution[2] in the following form:

$$
\begin{aligned}
Pr\Big[\chi_k^2 > (1+\epsilon)k\Big] &= Pr\Big[e^{\lambda\chi_k^2} > e^{\lambda(1+\epsilon)k}\Big] && \forall\ \lambda \geq 0 \\
&\leq \frac{\mathbb{E}\Big[e^{\lambda\chi_k^2}\Big]}{e^{\lambda(1+\epsilon)k}} && \text{using Markov Inequality} \\
&= \frac{(1-2\lambda)^{-k/2}}{e^{\lambda(1+\epsilon)k}} && \text{using MGF of } \chi_k^2 \\
&= \frac{(1+\epsilon)^{k/2}}{e^{\lambda(1+\epsilon)k}} && \text{choose } \lambda = \frac{\epsilon}{2(1+\epsilon)} \text{ to minimize failure} \\
&= \big((1+\epsilon)e^{-\epsilon}\big)^{\frac{k}{2}} \\
&\leq \exp\big(-\frac{k}{4}(\epsilon^2 - \epsilon^3)\big) && \text{using } 1+\epsilon \leq \exp\big(\epsilon - (\epsilon^2-\epsilon^3)/2\big)
\end{aligned}
$$

[1] Markov Inequality: $Pr[x \geq a] \leq \mathbb{E}[x]/a$

[2] Moment generating function of $X \sim \chi_k^2$: $M_X(t) = \mathbb{E}[e^{\lambda X}] = (1-2\lambda)^{-k/2}$

This proves the failure probability for $Pr[||R^T\boldsymbol{x}||^2 > (1+\epsilon)||\boldsymbol{x}||^2] \leq \exp\big(-\frac{k}{4}(\epsilon^2-\epsilon^3)\big)$ is bounded. The lower bound $Pr[||R^T\boldsymbol{x}||^2 < (1-\epsilon)||\boldsymbol{x}||^2]$ is proved in a similar manner. Therefore the probability of failure for $\boldsymbol{x_1}, \boldsymbol{x_2}$ is:

$$Pr\Big[||R^T\boldsymbol{x_1} - R^T\boldsymbol{x_2}||^2 > (1+\epsilon)||\boldsymbol{x_1}-\boldsymbol{x_2}||^2\Big] \bigcup$$
$$Pr\Big[||R^T\boldsymbol{x_1} - R^T\boldsymbol{x_2}||^2 < (1-\epsilon)||\boldsymbol{x_1}-\boldsymbol{x_2}||^2\Big] \leq 2\exp\big(-\frac{k}{4}(\epsilon^2-\epsilon^3)\big)$$

It is worth noting that this proof shows that nothing is fundamental about using Gaussian in particular. Many distributions with unit variance and certain boundedness properties (or higher order moment conditions) suffice.

A.3 Step 3. Bound the failure probability for all pairs of points

In the previous step we were able to bound the failure probability for 2 points $\boldsymbol{x_1}, \boldsymbol{x_2}$. In the final step, we would like to bound the failure probability for $\boldsymbol{x_i}, \boldsymbol{x_j}$ $i, j \in [n],\ i \neq j$, any two points in the dataset. For this, we refer to Bonferroni's Union Bound which states

$$Pr\Big[\bigcup E_i\Big] \leq \sum Pr\Big[E_i\Big]$$

where E_i represents a probability event and i ranges from 1 to the number of events in consideration. In this case, the dataset contains n distinct points, and hence $\frac{n(n-1)}{2}$ distinct pairs of points, which is the number we enumerate. Therefore

$$\begin{aligned} Pr\Big[\text{failure for } \textit{any} \text{ two points}\Big] &\leq \frac{n(n-1)}{2} \times 2\exp\big(-\frac{k}{4}(\epsilon^2-\epsilon^3)\big) \\ &= n(n-1)\exp\big(-\frac{k}{4}(\epsilon^2-\epsilon^3)\big) \\ &\leq \delta \end{aligned}$$

where we have shown that δ can be chosen based on the tolerance ϵ, number of samples in the point-set n, and the embedding dimension k. □