Small Project - Dimensionality Reduction and Random Projections

Group 9: David Nordmark, Paulo Hernández, Axel Drotz, Erik Karlström[*]

Dated: January 9, 2022

Abstract

With more data becoming available, dimensionality reduction is getting more common in many areas that handles information. There are many ways to reduce the dimensionality of data and it is useful to know how these different techniques work and which you should use for your purpose. This paper present experimental results on the qualities of four techniques used in dimensionality reduction and compares the results to the findings of a previously released scientific paper.

[*] Electronic address: dnordm@kth.se, pauloh@kth.se, axeldr@kth.se, ekarlstr@kth.se

1 Introduction

This paper tries to reproduce the findings of the article ”Random projection in dimensionality reduction: Applications to image and text data” by Ella Bingham and Heikki Mannila [1], their article will be referenced as the ”original” article throughout this paper.

In their article, Bingham and Mannila talks about dimensionality reduction and specifically argues for the benefits of using Random Projections (RP) for reducing dimensions. The article presents results for the computational complexity and the amount of distortion caused by RP and compares its results with other methods used in dimensionality reduction, such as the more conventional method, Principal Component Analysis (PCA), which is the optimal way to project data onto lower dimensions in the mean-square sense. However, PCA is computationally expensive, that is why the authors argue, with the support of their results, that random projections is a useful method for dimensionality reduction of computationally demanding data.

In this paper, the methodology behind dimensionality reduction is explained and results regarding similarity preservation and time complexity of the four different techniques are presented. The results include how these methods depend on the chosen dimension. At the end of this paper, the results are discussed and compared to the previous findings of Bingham and Mannila.

2 Methods

The four techniques/methods that were used in the original article were re-implemented and used to reduce the dimensions of the same image and text data sets also used in the original article. The data sets that were used in the original article will be referenced throughout this paper as the ”original data sets”. The original image data set consists of 13 gray scale images of natural scenes [11], from which a total of 1000 frames of size 50×50 were randomly obtained. As described in the article, the frames were then represented in a 1×2500 array. By using the same methods and data sets as the original article, we hoped to get a reproduced results which includes all original features.

In addition to the original data sets, another data set was used to ensure that the results are reproducible with other data sets, not solely the one used in the original article. The supplementary data set included images of airplanes [2], and just like in the original article, these images were gray scaled and windows were randomly drawn from the images with a size of 50×50. Each airplane image was then represented as a 1×2500 array using the pixels from the drawn window. These one dimensional arrays were collected as columns in a matrix. Each image yielded a one dimensional array and there was 800 images in total. The finished data set was therefore a 2500×800 matrix. One important factor to mention is that each data vector was normalized by applying Equation 1:

$$\hat{u} = \frac{u}{|u|} \tag{1}$$

The article also makes use of a text data set, which consists on a group of text documents, which were converted into term frequency vectors. As the authors imply, this text data ”was not made zero mean, nor was the overall variance of entries of the data matrix normalized” [1], it only was normalized to unit length. At the end, the text data had a shape of 5,000 terms and 2262 documents. All data set matrices (the original, the new airplane images and the text data sets) were then exposed to dimensional reduction using the four different techniques. All four techniques yielded different lower-dimensional spaces. These differences were compared on how well they preserved similarities of the data sets after dimensional reduction. This was done by calculating the Euclidean distance or the Inner Dot Product between two data vectors before and after the dimensional reduction and comparing how well they have scaled appropriately. It is important to point out that such vectors consisted of a sample of 100 different pairs picked randomly. The error was calculated using the formula shown in Equation 2.

$$error = \frac{(V_{obtained} - V_{expected})}{V_{expected}} \tag{2}$$

In addition, the computation time for the different methods was calculated and compared. The computation time was calculated using real time, i.e. how many seconds it took for a method to finish its dimensionality reduction of a data set, this is different from the original article where they calculated computation time using flops as a way of measuring computer performance [3], but it is still expected to obtain the same features and behaviour in the results. This decision was taken under the fact that "complexity is defined by the execution time and memory resources required to perform the computation", as defined by Tom Mens in his article "Research trends in structural software complexity" [6].

2.1 The four techniques

SRP and RP

Random projection, *RP*, and Sparse random projection, *SRP*, are random projections, which takes original data X of some dimensions d × N and maps that to a lower dimension k × N. This is done using a randomly generated matrix R of size k × d, were R is a linear mapping. In RP, each element in the R matrix is selected randomly from a Gaussian distribution with mean 0 and standard deviation 1. Each column is later set to unit length.

In SRP each element in R is selected at random as in Equation 3.

$$r_{ij} = \sqrt{3}\begin{cases} +1 \texttt{ with probability } \frac{1}{6} \\ 0 \texttt{ with probability } \frac{2}{3} \\ -1 \texttt{ with probability } \frac{1}{6} \end{cases} \tag{3}$$

After constructing this linear mappings, R, the original data is mapped to a lower dimensional by the matrix multiplication, R × X. [1, 7]

Principal Component Analysis

Principal components is the most common method for dimensionality reduction since it performs a linear mapping in such a way that it minimizes the average squared distance from the data points to the reduced dimension. I.e., PCA finds the most important dimensions for the data and projects onto these. This is useful since the dimensions that are lost during dimensionality reduction are the most negligible dimensions of the data, which results in a minimal loss of variation in the mean-square sense. However, the drawback of performing PCA is that it is rather computationally demanding. [1, 4]

Discrete Cosine Transform

DCT is a technique widely used in data compression and can be used in images, video, audio and more. It is commonly used in file compression formats such as jpeg. DCT uses cosines to transform a finite data set into frequencies, comparable to discrete Fourier transformation but while only using real values. Therefore, DCT can be used to reduce the dimensions of image data, even being independent of the data which results in much lower computational complexity than PCA. [1, 5]

2.2 The error calculation

2.2.1 Error Calculation for image data

The euclidean distance for the original images was calculated as: $||x_i - x_j||$, for a vector pair $[i, j]$. This is what we will refer to as $V_{expected}$. The error for each method RP, SRP, PCA, DCT was calculated slightly differently which will be explained in this section. The final calculation to obtain the error is mentioned in equation (2), what differs in all methods is how $V_{obtained}$ is calculated.

Error calculation for SRP and RP
For SRP and RP the euclidean distance after dimensional reduction was calculated as.

$$V_{obtained} = \sqrt{\frac{d}{k}}||Rx_i - Rx_j|| \tag{4}$$

Where R is the matrix used for dimensional reduction and x_i, x_j are two different image vectors. R is a k × d size matrix, and each vector i is of size $d \times 1$. Due to the dimensional reduction a scaling factor was introduced, this scaling factor is $\sqrt{\frac{d}{k}}$. The scaling factor was only used for RP and SRP.

Error calculation for PCA
PCA is an eigenvalue decomposition of the covariance matrix computed as $E\{XX^T\} = E\Lambda E^T$. The linear mapping to lower dimension can then be calculated as Equation 5.

$$X^{PCA} = E_k^T X \tag{5}$$

From here the euclidean distance was calculated as Equation 6.

$$V_{obtained} = ||E_k^T X_i - E_k^T X_j|| \tag{6}$$

Error calculation for DCT
DCT transforms a finite data set into frequencies. The first step is to transform the data into frequencies. When we have the data in cosine-space we remove the frequencies that are of least significance. Then we transform it back, with a basis-matrix of smaller dimensionality to match the new reduced vector. After the transformation we use the same method for the error as SRP and RP, but without the constant of $\sqrt{\frac{d}{k}}$.

2.2.2 Error calculation for Text Data

When verifying the effectiveness of the dimensional reduction algorithms on the text data, the authors make a special emphasis that for this specific kind of data the error is calculated as the inner dot product of two vectors. This is applied under the premise that each of the input vectors has been normalized to unit length. The inner dot product between 2 vectors can therefore be geometrically interpreted ”as the length of the projection of the first unit vector onto the second unit vector” [10].

3 Results

In this section, all the results will be presented for the different tests performed. In order to verify the veracity of the algorithm implemented, the first data set to be subjected to the run of the algorithms was the original data set that consists of monochrome images of natural scene. As mentioned above, the original data set is represented through a matrix size 2500 × 1000, where the columns represent each single randomly drawn 50 × 50 frame out of the 13 nature images. The new dimensions of which the data was reduced to, are shown in Equation 7.

$$k = 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 40, 60, 80, 100, 150, 200, 300, 500, 750, 1000 \tag{7}$$

Figure 1, presented below, shows the yielded error by each of the algorithms when subjected to each one of the dimensional reductions, according to the calculation shown in Equation 4. Apart from the error yielded by each of the algorithms, the article also poses an interest on the computational complexity of the different algorithms, as discussed before. The obtained execution time results for the original data set are presented as well in the right side of Figure 1.

Once, the set of algorithms were tested and compared against the results presented in the work of Bingham and Mannila, the tests were then performed on the newly formed data set of airplane

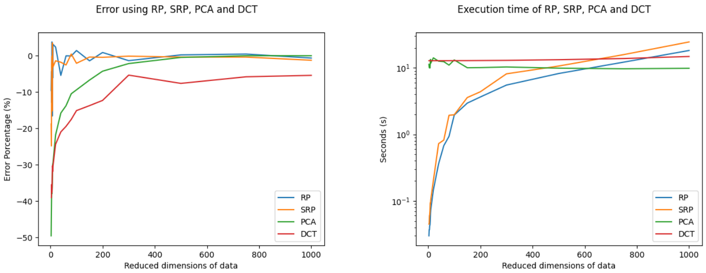

Figure 1: The average error (left) and execution time (right) produced by RP, SRP, PCA and DCT on the original image data over 100 pairs of data vectors.

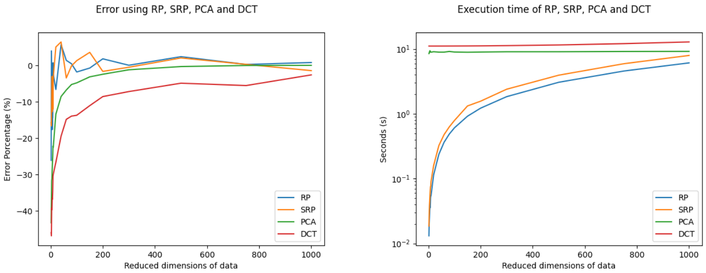

Figure 2: The average error (left) and execution time (right) produced by RP, SRP, PCA and DCT on the newly created image data over 100 pairs of data vectors.

images. The error produced by each algorithm across the different k dimensions, as well as the respective execution time is shown below on Figure 3.

Once finished with the image analysis, then the text data set is subjected to the same procedure. It is important to recall that the text data subjected to the dimensional reduction techniques has an inherent shape of 5000 × 2000. The results for both the average error and execution time are shown below on Figure 3.

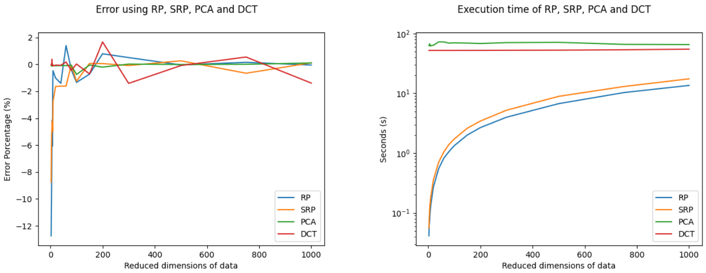

Figure 3: The average error (left) and execution time (right) produced by RP, SRP, PCA and DCT on the text data over 100 pairs of data vectors.

4 Discussion

4.1 The error calculation

In the original article it is not clearly stated how the error is calculated. Our assumption is that the error is calculated as in equation 1. The problem that arises is that for each data dimension k there are produces 100 vector pairs which the error is averaged over. This means that the error can be described as

$$avg_{error} = \frac{1}{100}\sum_{n=0}^{100}\frac{(V_{n,obtained} - V_{n,expected})}{V_{n,expected}} \tag{8}$$

This is to our understanding how the error has been calculated in [1], this means that some terms in this sum negate each other as some values for n might produce a positive value and others values might produce a negative value. Therefore we thought it would be more suitable to calculate the error as

$$avg_{error} = \frac{1}{100}\sum_{n=0}^{100}|\frac{(V_{n,obtained} - V_{n,expected})}{V_{n,expected}}| \tag{9}$$

In this equation the absolute sign ensures that only the magnitude of the error is taken to account as the sign of the error is not of importance. This equation would yield a worse result with a greater average error.

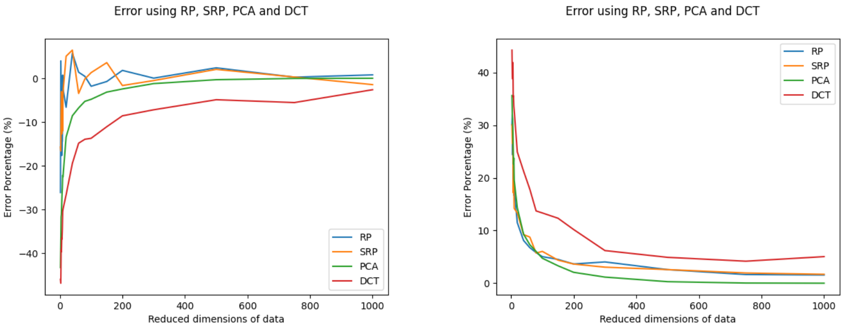

Figure 4: Comparison between average error (left) and absolute average error (right) produced by RP, SRP, PCA and DCT on the newly created image data over 100 pairs of data vectors.

In principal component analysis (PCA), the reduced dimension is spanned by the most important eigenvectors of the data set. If the preservation of mean-square distance is prioritized, you should definitely consider using PCA even though it is computationally demanding. However, the results show that for our high dimensional data set, PCA yields a larger error when projecting onto small k-dimensions than for example RP, regardless of its longer computation time (see Figure 1-3). Only after $k \approx 600$ do we see accurate results for PCA. This shows that while PCA is an optimal way to reduce dimensions in one sense, it can still yield larger error than other methods depending on the situation. Knowing when and if you can afford to use PCA for ”optimal” results is important.

4.2 DCT Computation Time

We notice that the DCT method has longer computation time than advertised, shown in Figure 1-2. The reason for this is that when using a package the basis vectors are already pre-computed, while we have to compute the entire set for the compression for every decrease in dimensionality. Without having to compute this matrix, it is only a matter of simple matrix multiplication, which is much faster than computing a matrix of dimensions k x k. See Figure 5.

4.3 Data Preprocessing

One important issue with the reproduction of these tests was the data preprocessing stage. Just as the authors describe in their article, ”differences in preprocessing yielded slightly different results on these different data sets” [1]. In the case of the image data, it´s never mentioned if the vectors should be normalized to unit length, making it difficult to understand how should the input data matrix be constructed. At the end, it was a matter of trial and error until the conclusion was drawn that each data vector should be normalized according to Equation 1, for better results.

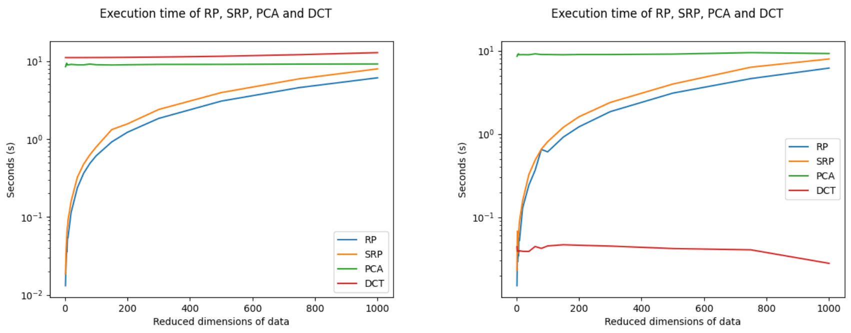

Figure 5: The difference in computation time with (left) or without (right) computing the basis vectors.

4.4 Reproducibility

The original article had high reproducibility since they described their methodology and results well enough for us to be able to replicate the study, even using the same data set, and getting very similar results. Being able to reproduce the findings of scientific papers is very important for the validity of the paper and in turn scientific progress. The results of the original article could also get generalized to the new data set that we tested.

5 Conclusion

The conclusion of the original article is that ”random projection preserves the similarities of the data vectors well even when the data is projected to moderate numbers of dimensions; the projection is yet fast to compute” [1]. Their conclusion cohere with our result, but since the original article was published, new methods for dimensionality reduction have been found and more research on the subject has been made. Today, random projections are not the most common/popular method for dimensionality reduction, but they can still become useful in certain situations [8, 9].

From this project we have learned that random projections in its simplicity works rather well. While keeping the computation time short, it still produces accurate results. While PCA and methods that involve eigen-decomposition are superior in the sense that the matrices can be used for more than just dimensionality-reduction once we have them computed, they take longer to execute. For the pure purpose of getting a good result fast, random projections work well enough.

We have also learned the methodology of jpeg and such formats, by attempting the Quantification-matrix while performing DCT. While not necessary, it was good to get some insight into the area.

References

[1] [Bingham and Mannila, 2001] Bingham, E. and Mannila, H. (2001). Random projection in dimensionality reduction: applications to image and text data. In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, pages 245–250.

[2] L. Fei-Fei, R. Fergus and P. Perona. Learning generative visual models from few training examples: an incremental Bayesian approach tested on101 object categories. IEEE. CVPR 2004, Workshop on Generative-ModelBased Vision. 2004. [online] Available at:
<http://www.vision.caltech.edu/Image_Datasets/Caltech101/#Download>

[3] En.wikipedia.org. 2022. FLOPS - Wikipedia. [online] Available at:
<https://en.wikipedia.org/wiki/FLOPS>

[4] En.wikipedia.org. 2022. Principal component analysis - Wikipedia. [online] Available at:
<https://en.wikipedia.org/wiki/Principal_component_analysis>

[5] En.wikipedia.org. 2022. Discrete cosine transform - Wikipedia. [online] Available at:
<https://en.wikipedia.org/wiki/Discrete_cosine_transform>

[6] Mens, Tom. 2016. Research trends in structural software complexity. [online] Available at:
<https://www.researchgate.net/publication/305857637_Research_trends_in_structural_software_complexity>

[7] S, S., n.d. Random Projection in Dimensionality Reduction. [online] Machine Learning Medium. Available at:
https://machinelearningmedium.com/2017/07/28/random-projection-in-dimensionality-reduction/

[8] Sharma, P., 2018. Dimensionality Reduction Techniques — Python. [online] Analytics Vidhya. Available at:
<https://www.analyticsvidhya.com/blog/2018/08/dimensionality-reduction-techniques-python/>

[9] Pramoditha, R., 2021. 11 Dimensionality reduction techniques you should know in 2021. [online] Medium. Available at:
<https://towardsdatascience.com/11-dimensionality-reduction-techniques-you-should-know-in-2021-dcb9500d388b>

[10] Weisstein, Eric W. Dot Product. [online] MathWorld–A Wolfram Web Resource. Available at:
<https://mathworld.wolfram.com/DotProduct.html>

[11] Helsinki University of Technology, 1997. Natural Image Collection for ICA experiments. [online] Available at: <https://web.archive.org/web/20150412005848/https://research.ics.aalto.fi/ica/data/images/>