RANGE-NET: A HIGH PRECISION NEURAL SVD

Anonymous authors
Paper under double-blind review

ABSTRACT

For Big Data applications, computing a rank-r Singular Value Decomposition (SVD) is restrictive due to the main memory requirements. Recently introduced streaming Randomized SVD schemes work under the restrictive assumption that the singular value spectrum of the data has an exponential decay. This is seldom true for any practical data. Further, the approximation errors in the singular vectors and values are high due to the randomized projection. We present Range-Net as a low memory alternative to rank-r SVD that satisfies the lower bound on tail-energy given by Eckart-Young-Mirsky (EYM) theorem at machine precision. Range-Net is a deterministic two-stage neural optimization approach with random initialization, where the memory requirement depends explicitly on the feature dimension and desired rank, independent of the sample dimension. The data samples are read in a streaming manner with the network minimization problem converging to the desired rank-r approximation. Range-Net is fully interpretable where all the network outputs and weights have a specific meaning. We provide theoretical guarantees that Range-Net extracted SVD factors satisfy EYM tail-energy lower bound with numerical experiments on real datasets at various scales that confirm these bounds. A comparison against the state-of-the-art streaming Randomized SVD shows that Range-Net is six orders of magnitude more accurate in terms of tail energy while correctly extracting the singular values and vectors.

1 INTRODUCTION

Singular Value Decomposition (SVD) is pivotal to exploratory data analysis in identifying an invariant structure under a minimalistic representation (assumptions on the structure) containing the span of resolvable information in the dataset. Finding a low rank structure is a fundamental task in applications including Image Compression (de Souza et al., 2015), Image Recovery (Brand, 2002), Background Removal (Wang et al., 2018), Recommendation Systems (Zhang et al., 2005) and as a pre-processing step for Clustering (Drineas et al., 2004) and Classification (Jing et al., 2017). With the advent of digital sensors and modern day data acquisition technologies, the sheer amount of data now requires that we revisit the solution scheme with reduced memory consumption as the target. In this work, we reformulate SVD with special emphasis on the main memory requirement, with no loss in accuracy, that precludes it's use for big data applications.

It is well known that natural data matrices have a decaying spectrum wherein saving the data in memory in its original form is either redundant or not required from an application point of view. However, any assumption on the decay rate can only be validated if the singular value decomposition is known *a priori*, which is seldom the case in exploratory data analysis (see **Fig. 3**). Visually assessing a rank-r approximation for image processing applications might seem correct qualitatively (see **Fig. 4**) but are still prone to large errors due to limited human vision acuity (see **Fig. 6**). This is further exacerbated when the application at hand is associated with scientific computations wherein the anomalies or unaccounted phenomena are still being explored from large scale datasets. The reader is preemptively referred to **Fig. 19** where the high frequency features related to turbulence cannot be disregarded. Furthermore, for classification and clustering problems where feature dimension reduction is desirable it is imperative that a low-rank approximation of a dataset contains most ($\geq 90\%$) of the original information content without altering the subspace information. In this case, an over-sampled rank can exceed the feature dimension of the data itself (see **Section 4.2**).

1.1 PROBLEM STATEMENT

Let us denote the raw data matrix as $X \in \mathbb{R}^{m\times n}$ of rank $f \leq \min(m, n)$ and its approximation as $X_r \in \mathbb{R}^{m\times n}$. The SVD of X is $X = U\Sigma V^T$, where $U \in \mathbb{R}^{m\times f} = [u_1, \cdots, u_f]$ and $V \in \mathbb{R}^{n\times f} = [v_1, \cdots, v_f]$ are its left and right singular vectors respectively, and $\Sigma \in \mathbb{R}^{f\times f} = diag(\sigma_1, \cdots, \sigma_f)$ are its largest non-zero singular values. The rank r $(r \leq f)$ truncation of X is then $X_r = U_r\Sigma_r V_r^T$, where $\Sigma_r = \Sigma_{[1:r]}$ are the top r singular values, and $U_r = U_{[1:r]}$ and $V_r = V_{[1:r]}$ are the left and right singular vectors. In other words, $X = U\Sigma V^T = U_r\Sigma_r V_r^T + U_{f\setminus r}\Sigma_{f\setminus r}V_{f\setminus r}^T = X_r + X_{f\setminus r}$. Here, $U_{f\setminus r}, V_{f\setminus r}$ are the trailing $f - r$ left and right singular vectors.

Theorem 1. *Eckart-Young-Mirsky Theorem (Eckart & Young, 1936; Mirsky, 1960): Let $X \in \mathbb{R}^{m\times n}$ be a real, rank-f, matrix with $m \geq n$ with the singular value decomposition as $X = U\Sigma V^T$, where the orthonormal matrices U, V contain the left and right singular vectors of X and Σ is a diagonal matrix of singular values. Then for an arbitrary rank-r, $r \leq f$ matrix $B_r \in \mathbb{R}^{m\times n}$,*

$$\|X - B_r\|_F \geq \|X - X_r\|_F$$

where $X_r = U_r\Sigma_r V_r$ with Σ_r is the diagonal matrix of the largest r singular values and U_r, V_r are the corresponding left and right singular vector matrices.

The problem statement is then: Given $X \in \mathbb{R}^{m\times n}$ find $\hat{X}$ such that,

$$\underset{\hat{X}\in\mathbb{R}^{m\times n},\ \mathrm{rank}(\hat{X})\leq r}{\arg\min} \|X - \hat{X}\|_F \tag{1}$$

In effect, the minimizer $\hat{X}_*$ of the above problem gives us the rank-r approximation of X such that $X_r = \hat{X}_*$. In this work we utilize the minimizer of the above problem to extract the top rank-r SVD factors of X without loading the entire data matrix into the main memory. *Note that the minimizer naturally gives the lower bound on this tail energy in addition to being a rank-r approximation.*

1.2 MAIN CONTRIBUTIONS

Data and Representation Driven Neural SVD: The representation driven network loss terms ensures that the data matrix X is decomposed into the desired SVD factors such that $X = U\Sigma V^T$. In the absence of the representation enforcing loss term, the minimizer of Eq. 1 results in an arbitrary decomposition such that $X = ABC$ different from SVD factors.
A Deterministic Approach with GPU Bit-precision Results: The network can be initialized with weights drawn from a random distribution where the iterative minimization is deterministic. The streaming order of the samples is of no consequence and the user is free to choose the order in which the samples are processed in a batch-wise manner (indexed or randomized).
First Streaming Architecture with Exact Low Memory Cost: Range-net requires an exact memory specification based upon the desired rank-r and data dimensions $X \in \mathbb{R}^{m\times n}$ given by $r(n + r)$, independent of the sample dimension m. This is the first streaming algorithm that does require the user to wait until the streaming step is complete, contrary to randomized streaming algorithms.
Layer-wise Fully Interpretable: Range-Net is a low-weight, fully interpretable, dense neural network where all the network weights and outputs have a precise definition. The network weights are placeholders for the right (or left) orthonormal vectors upon convergence of the network minimization problems (see **Appendix D**). The user can explicitly plug a ground truth solution to verify the network design and directly arrive at the tail energy bound.

2 RELATED WORKS

The core idea behind randomized matrix decomposition is to make one or more passes over the data and compute efficient sketches. They can be broadly categorized into four branches: 1) Sampling based methods (Subset Selection (Boutsidis et al., 2014) and Randomized CUR (Drineas et al., 2006)); 2) Random Projection based QR (Halko et al., 2011); 3) Randomized SVD (Halko et al., 2011); and 4) Power iteration methods (Musco & Musco, 2015). The sketches can represent any combination of row space, column space or the space generated by the intersection of rows and columns (core space). However, all of these methods require loading the entire data in memory. Readers are referred to Kishore Kumar & Schneider (2017); Ye et al. (2019) for an expanded survey.

Conventional SVD although deterministic and accurate, becomes expensive when the data size increases and requires r passes over the data (see **Table 1**). The two branches of interest to us are the randomized SVD and Power iteration Methods for extracting SVD factors. Randomized SVD algorithms (Halko et al., 2011) are generally a two stage process: **1) Randomized Sketching** uses random sampling to obtain a reduced matri(x/ces) which covers any combination of the row, column and core space of the data; and **2) Deterministic Post-processing** performs conventional SVD on the reduced system from Randomized Sketching stage. These approaches make only one pass over the data assuming that the singular value spectrum decays rapidly.

Power iteration based approach (Musco & Musco, 2015) requires multiple passes over the data and are used when the singular value spectrum decays slowly. This class of algorithm constructs a Krylov matrix inspired by Block Lanczos (Golub & Underwood, 1977) to obtain a polynomial series expansion of the sketch. Although these algorithms achieve lower tail-energy errors, they cannot be used in big-data applications when X itself is too large to be retained in the main memory. Here, constructing a Krylov matrix with higher order terms such as (AA^T or A^TA) is not feasible[1].

Table 1: Current best randomized SVD Methods. k, l, s are overestimated sketch sizes for a rank-r estimate *s.t.* $k, l, s > r$. Note that Range-Net has an exact memory requirement (as conventional SVD), unlike order bounded Randomized methods. Note that deterministic implies the solution obtained upon convergence is deterministic.

Method	Halko et al. (2011)	Upadhyay (2016)	Tropp et al. (2017b)	Tropp et al. (2019)	Range-Net	Conventional SVD
Space Complexity	$\mathcal{O}(k(m+n))$	$\mathcal{O}(k(m+n)+s^2)$	$\mathcal{O}(km+nl)$	$\mathcal{O}(k(m+n)+s^2)$	$r(n+r)$	$n(m+2n)$
# Passes	1	1	1	1	≤ 5	r
Type	Randomized	Randomized	Randomized	Randomized	Deterministic	Deterministic

Due to main-memory restrictions on remote compute machines, streaming (Clarkson & Woodruff, 2009) algorithms became popular. For low-rank SVD approximations these involve streaming the data and updating low-memory sketches covering the row, column and core spaces. Existing randomized SVD capable of streaming include Halko et al. (2011); Upadhyay (2016); Tropp et al. (2017a; 2019), each with different sketch sizes and upper bounds on approximation errors (**Table 1**).

SketchySVD (Tropp et al., 2019) is the state of the art streaming randomized SVD, with sketch sizes comparable to it's predecessors and tighter upper bounds on the tail energy and lower errors. As a two stage approach, SketchySVD (**Alg. 1**) constructs an overestimated rank-(k, s) sketch of the data based on row, column and core projections. A QR decomposition on the row and column sketches gives an estimate of the rank-k subspace. This is followed by a conventional SVD on the core matrix to extract it's singular values and vectors. Finally, the singular vectors are returned after projecting them back to the original row and column space. The time cost of SketchySVD is $\mathcal{O}(k^2(m+n))$ with memory cost $\mathcal{O}(k(m+n)+s^2)$ with oversamling parameters $k = 4r+1$ and $s = 2k+1$.

2.1 Limitations of Randomized SVD Approaches

We discuss a few limitations that led us to reformulate the problem in the spirit of EYM theorem. The reader is referred to **Appendix A** for a detailed discussion and supporting numerical examples to further enunciate these limitations.

Tall and Skinny Matrices: For a rank-r approximation of $X \in \mathbb{R}^{m\times n}$ of rank-f, Randomized SVD methods rely upon rank-k ($k > r$) sketches of X. However, these methods are useful only when $k \geq f$ but for practical datasets $f \leq \min(m, n)$ and therefore the memory requirement can still be overbearing. The reader is referred to **Section 4.2** for a practical example.
Exponential Decay of Singular Values: Assuming exponential decay implies the rank of X itself is such that $f \ll \min(m, n)$. For real world applications, the data matrices are almost full rank $f \leq \min(m, n)$, where a rank r truncation is chosen such that the desired dominant features are accounted for. **Appendix A** shows a synthetic case with non-exponential decay of singular values where sketching accrues substantial errors. Further, it is difficult to assume that the decay rate will follow a strict functional form: mixture of linear, exponential and others (see **Fig. 3**).
Upper Bound on Tail Energy: The problem statement in **Eq. 1** suggests finding a minimum with the minimizer providing a lower bound on the tail-energy. Even if the solution scheme is upper bounded (Halko et al., 2011; Upadhyay, 2016; Tropp et al., 2017a; 2019), the minimizer $\hat{X}_*$ in Eq. 1 or equivalently achieving the lower bound is necessary.

[1]Readers are referred to **Fig. 5 (c)** and **Fig. 8** for a performance comparison between power iteration schemes and Range-Net for a mid-sized real and a synthetic dataset that can be loaded on our compute machine.

Approximation Errors: A low-rank SVD solver that does not iteratively compute the projection (left or right) while solving **Eq. 1** cannot extract SVD factors with low errors even with multiple passes over the data matrix or multiple runs. As shown later in **Theorem 2**, any subspace projection (left or right) of the original data matrix that does not correspond to the minimizer in **Eq. 1** increases the tail energy and therefore results in incorrect low-rank SVD factors (singular values and vectors). **Memory Requirement:** Randomized SVD requires an optimal choice of hyper-parameters (sketch sizes *etc.*) that are subjective to the dataset being processed. In a practical, limited memory scenario, this entails tuning the hyper-parameters for optimal trade-off between memory requirement, compute time and an approximation error that does not violate the upper bound on the tail energy.

Remark. *A low relative error in tail energy does not imply the extracted singular values and vectors will have similar relative errors at scale. The issue has been raised by Musco & Musco (2015), that merely upper bounding the tail energy equipped with Frobenius or spectral norm does not bound the approximation errors in the extracted singular values or vectors. Therefore, for a fair comparison we show error metrics on the extracted singular factors for all our numerical experiments.*

3 RANGE-NET: A 2-STAGE SVD SOLVER

We present **Range-Net** that explicitly relies upon solving the minimization problem in **Eq. 1** to achieve the lower bound on the tail-energy for a desired rank-r approximation of a data matrix X under a streaming setting. The readers are referred to **Appendix B** for the preliminaries followed by proofs of theorems and lemmas associated with each of the two stages.

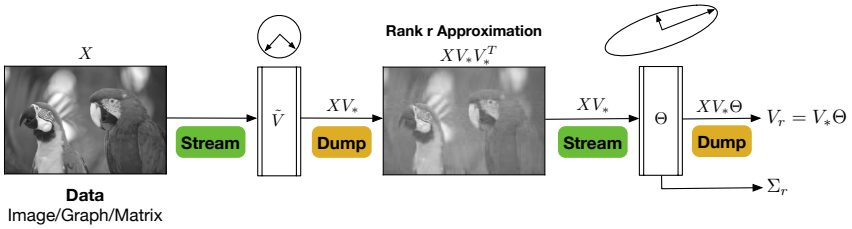


Figure 1: An overview of the low-memory, two-stage Range-Net SVD for Big Data Applications. **Stage 1** identifies the span of the desired rank-r approximation. **Stage 2** rotates this span to align with the singular vectors while extracting the singular values of the data. The input data can be streamed from either a server or secondary memory. If the target is just a rank-r compression then Stage-2 can be discarded without any loss in accuracy. Stage-2 only orders the rank-r features based upon their respective tail energies.

3.1 NETWORK ARCHITECTURE

The proposed network architecture is divided into two stages: (1) Projection, and (2) Rotation, each containing only one dense layer of neurons and linear activation functions with no biases. **Fig. 2** shows an outline of the this two-stage network architecture where all the weights and outputs have a specific meaning enforced using representation and data driven loss terms. Contrary to randomized SVD algorithms the subspace projection (Stage 1) is not specified preemptively (consequently no assumptions) but is computed by solving an iterative minimization problem following EYM theorem corresponding to Eq. 1. The rotation stage (Stage 2) then reuses the EYM theorem again in a modified form to extract the singular vectors and values.

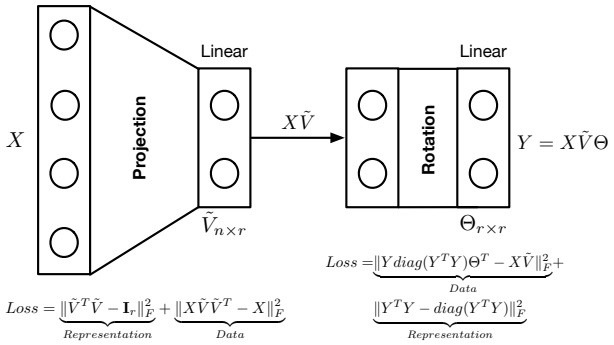


Figure 2: Network Architecture: Projection (Stage-1) and Rotation (Stage-2) for a 2-stage SVD

Stage 1: Rank-r Sub-space Identification: The projection stage constructs an orthonormal basis that spans the r-dimensional sub-space of a data matrix $X \in \mathbb{R}^{m\times n}$ of an unknown rank $f \leq min(m,n)$. This orthonormal basis ($\tilde{V}$) is extracted as the stage-1 network weights once the network minimization problem converges to a fixed-point. The representation loss $\|\tilde{V}^T\tilde{V} - I_r\|_F$ in stage-1 enforces the orthonormal requirement on the projection space (even when $r > f$) while the data-driven loss $\|X - X\tilde{V}\tilde{V}^T\|_F$ minimizes the tail energy. Although the minimization problem is non-convex, the tail-energy is guaranteed to converge to the minimum at machine precision. The reader is referred to **Appendix C** for a discussion on the minimization problem (bi-quadratic loss function with 2^r global minima) for details regarding the convergence behavior.

Theorem 2. *For any $r, f \in \mathbb{Z}^+$, $0 < r \leq f$, if the tail energy of a rank-f matrix $X \in \mathbb{R}^{m\times n}$, $f \leq min(m,n)$, with respect to an arbitrary rank-r matrix $B_r = X\tilde{V}_r\tilde{V}_r^T$ is bounded below by the tail energy of X with respect to it's rank-r approximation $X_r = XV_rV_r^T$ as, $\|X - B_r\|_F \geq \|X - X_r\|_F$ where, $V_r = span\{v_1, v_2, \cdots, v_r\}$ and v_is are the right singular vectors corresponding to the largest r singular values then the minimizer of $\underset{\tilde{V}\in\mathbb{R}^{(n\times r)}}{\arg\min} \|X - X\tilde{V}\tilde{V}^T\|_F$ is V_* such that $V_*V_*^T = V_rV_r^T$.*

As per **Theorem 2**, the equality holds true only when $span\{B_r\} = span\{X_r\}$. Further, we define B_r as, $B_r = X\tilde{V}\tilde{V}^T$ then, $V_*V_*^T = V_rV_r^T$ and $V_*^TV_* = I_r$ where V_r is a rank-r matrix with column vectors as top-r right singular vectors of X. The minimization problem then reads,

$$\min_{\tilde{V}} \quad \|X - X\tilde{V}\tilde{V}^T\|_F \quad s.t. \quad \tilde{V}^T\tilde{V} = I_r \tag{2}$$

with a minimum at the fixed point $V_* = span\{v_1, \ldots, v_r\}$ where $v_{i=1,\ldots,r}$ are the right singular vectors of X_r. This minimization problem describes the Stage 1 loss function of our network architecture. Upon convergence, the minimizer $\tilde{V}_*$ is such that $V_*V_*^T = V_rV_r^T$ following **Theorem 2** where V_r is the matrix with columns as right singular vectors of X corresponding to the largest r singular values of X.

Lemma 2.1. *If $V_r^TV_r = I_r$ and $V_rV_r^T = V_*V_*^T$ then $V_*^TV_* = I_r$.*

Lemma 2.2. *If $X \in \mathbb{R}^{m\times n}$ is a rank f matrix, then for any rank $r > f$, where $\{r, f\} \leq \min(m,n)$, if $V_*^TV_* = I_r$ and $V_*V_*^T = V_rV_r^T$ then $V_r^TV_r = I_r$.*

Remark. *Note that for $r \leq f$, the orthonormality constraint is trivially satisfied as shown in **Lemma 2.1**. However for $r > f$, the orthonormality constraint ensures that the column vectors in V_* are orthonormal (see **Lemma 2.2**) allowing us to extract orthonormal right singular column vectors of V_r from the Stage 2 minimization problem.*

Stage 2: Singular Value and Vector Extraction: The rotation stage then extracts the singular values by rotating the orthonormal vectors (V_*) to align with the right singular vectors ($V_r = V_*\Theta_r$). From the fixed point of the Stage-1 minimization problem Eq. 2 we have $V_*V_*^T = V_rV_r^T$. According to the EYM theorem the tail energy of a rank-r matrix XV_*C_r, where C_r is an arbitrary rank-r, real valued, square matrix, with respect to XV_* is now bounded below by 0 or $\|XV_* - XV_*C_r\|_F \geq 0$.

Theorem 3. *Given a rank-r matrix $XV_* \in \mathbb{R}^{m\times r}$ and an arbitrary, rank-r matrix $C \in \mathbb{R}^{m\times r}$, following **Theorem 1**, the tail energy of XV_* with respect to XV_*C is bounded as, $\|XV_* - XV_*C\|_F \geq 0$, where the equality holds true if and only if $C = I_r$.*

Lemma 3.1. *If $C = \Theta_r\Theta_r^T$, where $\Theta_r \in \mathbb{R}^{r\times r}$ is a rank-r matrix such that $C = I_r$, then Θ_r is a real-valued unitary matrix in an r-dimensional Euclidean space.*

Theorem 4. *Given a rank-r matrix $XV_* \in \mathbb{R}^{m\times r}$, such that $V_*V_*^T = V_rV_r^T$ where V_r is a matrix with column vectors as the top-r right singular vectors of X, and a real-valued unitary matrix $\Theta_r \in \mathbb{R}^{r\times r}$ then $(XV_*\Theta_r)^T(XV_*\Theta_r)$ is a diagonal matrix Σ_r^2 where $\Sigma_r^2 = \text{diag}(\sigma_1^2, \sigma_2^2, \cdots, \sigma_r^2)$ and σ_is are the top-r singular values of X if and only if $V_*\Theta_r = V_r$.*

From **Theorem 3** and **Lemma 3.1** we know that, $C_r = \Theta_r\Theta_r^T$, where Θ_r is a rank-r, unitary matrix in an r-dimensional Euclidean space. Further, from **Theorem 4** we have that $(XV_*\Theta_r)^T(XV_*\Theta_r)$ is a diagonal matrix $\Sigma_r^2 = \text{diag}(\sigma_1^2, \cdots, \sigma_r^2)$, where σ_is are the top-r singular values of X if and only if $V_*\Theta_r = V_r$. Assuming $Y = XV_*\Theta_r$, the minimization problem now reads:

$$\min_{\Theta_r} \quad \|Y\Theta_r^T - XV_*\|_F \quad s.t. \quad Y^TY - \text{diag}(Y^TY) = 0 \tag{3}$$

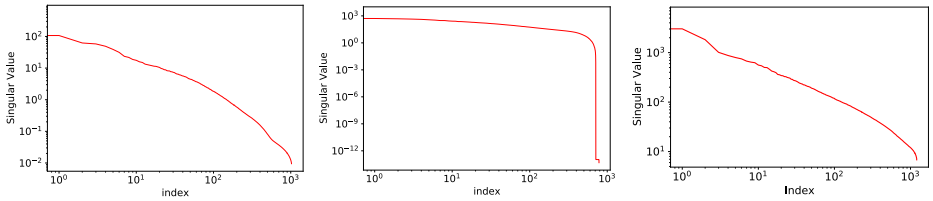
Remark. *Note that stage 1 can be verified numerically independently of stage 2 by checking whether the orthonormality condition is met in addition to minimization problem converging to the tail-energy bound. Similarly, stage 2 minimization problem will return a rotation matrix* Θ_r $(\Theta_r^T\Theta_r = \Theta_r\Theta_r^T = I_r, det(\Theta_r) = \pm 1)$ *upon convergence that can again be verified numerically.*

As discussed previously, this choice of loss terms equipped with a Frobenius norm ensures a rank-r approximation in accord with the Eckart-Young-Mirsky (EYM) theorem. We therefore state that the expected values of the stage-1 loss term at the minimum must correspond to the rank $(n - r)$ tail energy. Further, the stage-2 loss is expected to reach a machine precision zero at the minimum. Once, the network minimization problems converge, the singular values are extracted from **Stage 2** network weights Θ_r as $\Sigma_r^2 = (XV_*\Theta_r)^T(XV_*\Theta_r)$. The right singular vectors can now be extracted using **Stage 2** layer weights given by $V_r = V_*\Theta_r$. Once V_r and Σ_r are known, left singular vectors $U_r = XV_*\Theta_r\Sigma_r^{-1}$. Please note that for $r > f$, $f - r$ singular values are zero and therefore Σ_r^{-1} implies inverting the non-singular values using a threshold of 10^{-8}.

Implementation Details: A detailed discussion and justification of our implementation choices are given in **Appendix E**. These include *a) Activation*: all the activation functions are linear with no biases, since SVD requires linearly separable orthogonal features; *b) Data Split*: we do not perform any split of the training data, since SVD factors are unique to the full data only; *c) Data Streaming*: we stream data from HDD into main memory to avoid data sample load; *d) Training and Setup*; *e)* Error Metrics; and *f)* Loss Profiles. Note that Range-Net has no hyper-parameters and therefore does not require any post-hoc tuning or adjustments.

4 Results

We present numerical experiments for three datasets: (a) Parrot image, (b) MNIST, and (c) hurricane Sandy. The reader is referred to **Appendix E.5** for the definitions of the error metrics used. Note that, these error metrics rely upon conventional SVD as the baseline for a fair comparison. For additional numerical experiments on sparse graph datasets and other low rank approximations see **Appendix F**. SketchySVD's algorithmic implementation (**Alg. 1**) can be found in **Appendix G**.



(a) Parrot (b) MNIST (c) Sandy

Figure 3: Singular value spectrum for the three practical datasets considered in this work. One can visually assess that the decay rate of the singular values is non-exponential.

4.1 Image Compression: Parrots (SVD)

As an example for SVD of natural images, we use the well known Parrots image from the image processing domain. The original image is in an RGB format, converted to a gray scale for demonstration purposes followed by normalization between $[0, 1]$. This 1024×1536 data matrix is then used to compute a rank $r = 20$ approximation for comparison and numerical analysis. **Fig. 4** shows the result of the low rank reconstruction for SVD, SketchySVD and Range-Net. Visually one can verify that **Fig. 4 (b), (d)** are similar while **Fig. 4 (c)** is different. To make the error in approximation more clear, we plot the absolute difference of SketchySVD and our net from the truncated rank image. **Fig. 4 (e), (f)** shows the the corresponding plots with heatmaps imposed for clarity. Notice that while the reconstruction error for our network $(\approx 10^{-7})$ is close to the GPU precision, SketchySVD has significantly higher error scale $(\approx 10^{-1})$, validating the artifacts in the approximated image.

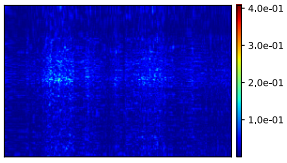
The singular value spectrum does not decay exponentially (**Fig. 3**) and the data matrix is near-full rank ($f \approx 1024$). **Fig. 5 (a, b)** shows the scree error as the absolute difference between the predicted and the true singular values. For SketchySVD, the error fluctuates across the top $r = 20$ values, but also the scale of fluctuations is around 1. Comparably, Range-Net incurs significantly lower errors in singular values (scale of 10^{-4}). **Fig. 5 (c)** shows the reconstruction errors in Frobenius norm for SketchySVD (Tropp et al., 2019) (red line), Block Lanczos with Power Iteration (Musco & Musco, 2015) (black line), Sklearn's randomized SVD (skr) implementation Halko et al. (2011) with (solid

(a) True Image

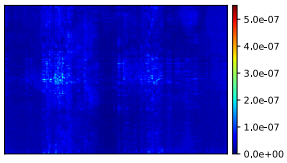
(b) SketchySVD $r = 20$

(c) Range-Net $r = 20$

(d) X_r for $r = 20$



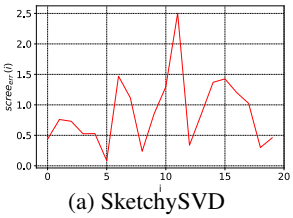
(e) $|\tilde{X}^{sketchy} - X_r|$



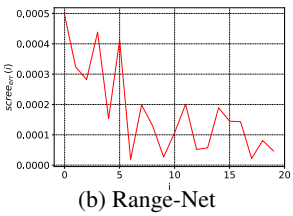
(f) $|\tilde{X}^{net} - X_r|$

Figure 4: (a) True image, rank-20 reconstruction using (b) conventional SVD, (c) SktechySVD (oversampled rank $k = 81$), (d) Range-Net (5-passes). Note that Sketchy SVD reconstruction error (10^{-1}) is 6 orders of magnitude apart from Range-Net's reconstruction error (10^{-7}).

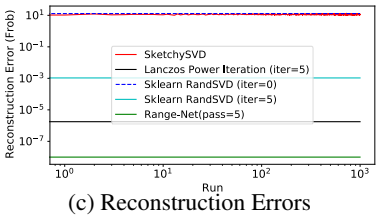
cyan line) and without power iteration (dashed blue line), and Range-Net (green line) over 1000 runs on the Parrot data. This shows that in order to gain lower reconstruction errors a power iteration is necessary that quickly becomes intangible in a big-data setting. Further, note that the expected error (upper bound) over multiple runs of Randomized SVD algorithms does not contract (reduce).



(a) SketchySVD



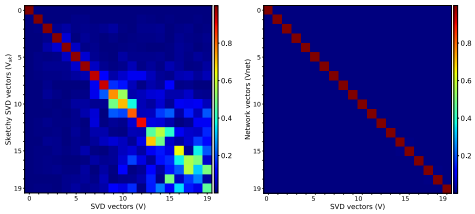
(b) Range-Net



(c) Reconstruction Errors

Figure 5: Scree error in the extracted singular values from (a) SketchySVD (≈ 1) and (b) Range-Net ($\approx 10^{-4}$). Notice the scale of errors. (c) Reconstruction errors (rank $r = 20$) for Range-Net and randomized SVD schemes (with and without power iterations) for Parrot image over 1000 runs.

Fig. 6 shows the cross-correlation between extracted right singular vectors from SketchySVD (left) and Range-Net (right) against conventional SVD for a rank-20 approximation. SketchySVD oversampled rank is $k = 81$ and still the extracted right singular vectors deviate substantially. This implies that the extracted vectors from SketchySVD do not span the top rank-20 subspace of X as opposed to Range-Net where stage-1 explicitly ensures this span without any oversampling. The higher the vector index, the higher the spread owing to random projections. Range-Net has a near-perfect cross-correlation with the true vectors, indicated by the solid diagonal and zero off-diagonal (near GPU-precision) entries.



(a) SketchySVD (b) Range-Net

Figure 6: Cross-correlation between true (conventional SVD) and extracted right singular vectors from (a) SketchySVD (b) Range-Net for a rank-$r = 20$ approximation of the Parrot image.

We tabulate error metrics (see **Appendix E** for definitions) for SketchySVD and Range-Net for various ranks in **Table. 2**. Note that all errors are reported *w.r.t.* the true SVD, where err_{fr} and err_{sp} denote the frobenius and spectral errors respectively. One can easily see that as the rank increases, SketchySVD's performance keeps on deteriorating for the χ^2_{err} metric (measure of correlation between true and estimated vectors), also evident from Fig. 6. Range-Net on the other hand has consistent lower errors in all metrics while simultaneously being memory efficient.

Table 2: Metric Performance and Memory of SketchySVD *vs.* Range-Net, for Parrot image

Rank	SketchySVD				RangeNet			
	err_{fr}	err_{sp}	χ^2_{err}	Mem (MB)	err_{fr}	err_{sp}	χ^2_{err}	Mem (MB)
$r = 10$	27.904	18.071	0.492	10.6	0.0	0.0	0.018	0.33
$r = 20$	11.974	2.201	0.662	21.69	0	0	0.023	0.69
$r = 50$	2.772	0.181	0.762	59.87	1.91e-7	0	0.027	1.72
$r = 100$	0.614	2.14e-2	0.923	139.9	2.32e-7	1.08e-7	0.033	3.6

4.2 DIMENSION REDUCTION: MNIST (EIGEN / PCA)

Principal Component Analysis (PCA) is a special variant of Eigen decomposition, where the samples are mean corrected before constructing a feature covariance matrix followed by Eigen decomposition. Note that Range-Net does not require construction of the feature covariance matrix and can directly extract the eigenvectors and values without any modification. This is due to the fact that the Eigenvalues are the square of the singular values for any non-square data matrix $X \in \mathbb{R}^{m \times n}$ where right singular vectors are exactly the same as the eigenvectors.

MNIST has 60k images of size 28×28 in the training set. We reshape each image into a 784-dim vector to obtain the data matrix $X \in \mathbb{R}^{60000 \times 784}$, as a *tall skinny matrix*. In a streaming setting, the mean feature vector computation requires one pass over the data matrix. This can be subsequently used during the network training (Stage-1) to mean correct streamed input vectors.

Table 3: Metric Performance and Memory of SketchySVD *vs.* Range-Net, for MNIST

Rank	SketchySVD				RangeNet			
	err_{fr}	err_{sp}	χ^2_{err}	Mem (GB)	err_{fr}	err_{sp}	χ^2_{err}	Mem (MB)
$r = 20$	1.09e3	8.29e2	1.34	0.47	0	0	0.012	0.51
$r = 50$	1.03e3	8.25e2	2.14	1.18	0	0	0.025	1.33
$r = 100$	1.05e3	8.47e2	2.03	2.38	1.12e-7	1.01e-7	0.052	2.83
$r = 200$	1.14e3	8.62e2	1.0	4.84	2.36e-7	2.52e-7	0.071	6.29

For this dataset, it is well known that $r = 200$ captures $\geq 90\%$ variance in the dataset. For SketchySVD, this results in projection matrices of ranks $k = 4r + 1 = 801$ and $s = 2k + 1 = 1602$. Since MNIST only has $n = 784$ features, SketchySVD (**Alg. 1**) is almost equal if not more memory intensive than conventional SVD for such tall and skinny matrices. **Table. 3** shows the error metrics under different rank setting, where even with oversampling, SketchySVD's errors are high. Range-Net on the other hand with an exact memory requirement $(r(n + r))$ can handle much larger full rank tall and skinny matrices without incurring extraneous memory load. As discussed before, since this data matrix is tall and skinny ($60k \times 784$) we already know that for SketchySVD any rank-r *s.t.* ($r \geq 196$) will result in the oversampling parameters $k \geq 784$ and $s \geq 1569$. SketchySVD will now extract lower-error SVD factors since the oversampled rank redundantly exceeds the feature dimension.

4.3 SCIENTIFIC COMPUTING: SANDY BIG DATA (SVD)

Satellite data gathered by NASA for Hurricane Sandy over the Atlantic ocean represents the big data counter-part for scientific computations. The data-set is openly available[2] and comprises of RGB snapshots captured at approximately one-minute interval. The full data-set consists of 896×719 pixel images for 1208 time-instances is of size 24 **GB**. We chose this particular big data so that a conventional SVD can be performed on our machine (16 GB RAM) for benchmarking. Please note that this restriction is imposed by conventional SVD method due to its high main memory requirements. In contrast, our neural SVD solver can handle data sets that are orders of magnitude larger in size with the same hardware specification.

Range-Net can not only handle larger datasets than SketchySVD, but also ensures lower errors in approximating the SVD factors. **Tab. 4** shows the error metrics for Range-Net with a comparison of peak main-memory load between SketchySVD and Range-Net for ranks $r = [10, 50, 100]$. **Appendix F.3** shows a comparison of dynamic mode reconstructions obtained from SketchySVD, conventional full rank SVD, and Range-Net and the associated scree errors in the computed singular values.

Table 4: Metric Performance and Memory of SketchySVD *vs.* Range-Net, for Sandy

Rank	SketchySVD				RangeNet			
	err_{fr}	err_{sp}	χ^2_{err}	Mem (GB)	err_{fr}	err_{sp}	χ^2_{err}	Mem (MB)
$r = 10$	1.43e3	7.72e2	0.47	2.56	0	0	0.011	0.39
$r = 50$	7.18e2	1.81e2	2.04	12.48	0	0	0.018	2.01
$r = 100$	4.32e2	6.68e1	3.022	24.91	1.12e-7	1.24e-7	0.021	4.19

Randomized SVD extracted factors deviate quite substantially when the user specified rank r is such that the oversampled rank k is much lower than the unknown rank f of a given data matrix. The reader is also referred to the additional experiment in **Appendix F.4** where a low rank ($r = 10$) approximation is extracted. Here, SketchySVD deviates quite substantially after rank-3 while Range-Net still remains in excellent agreement with the baseline singular vectors and values.

[2] https://www.nasa.gov/mission_pages/hurricanes/archives/2012/h2012_Sandy.html

4.4 STORAGE COMPLEXITY ANALYSIS

To estimate the memory efficiency of Range-Net, let us consider the peak main memory (RAM) requirement for the compuation of SVD factors. Range-Net has two layers in succession, one corresponding to the low-rank projector $\tilde{V}^{r\times n}$ and the rotation matrix $\Theta^{r\times r}$. For Sketchy SVD, the peak memory load occurs during the construction of a core matrix $C^{s\times s}$ (see **Alg. 1**). This requires that the two projection matrices $\Phi^{m\times s}, \Psi^{n\times s}$, one projected data matrix $Z^{s\times s}$, and two rank-k decomposition $Q^{m\times k}, P^{n\times k}$ and the core matrix $C^{s\times s}$, be present in the memory simultaneously. The memory efficiency factor (s_{eff}) for a rank-r approximation with $k = 4r + 1, s = 2k + 1$ is:

$$s_{eff} = \frac{\text{SketchySVD}}{\text{Range-Net}} = \frac{ns + ms + 2s^2 + mk + nk}{rn + r^2} = \frac{(m+n)(k+s) + 2s^2}{r(n+r)} \approx \frac{12(m+n) + 128r}{(n+r)}$$
$$\approx 7.67e2 \quad \text{for MNIST}(m = 60k, n = 784, r = 200)$$

To validate the ratio, we constructed a synthetic dataset of $m = 50k$ rows and the number of columns were varied starting at $n = 10k$ with increments of $10k$. The expected rank was held at $r = 200$. **Fig. 7** shows the memory allocation (in MegaBytes (MB)) of SketchySVD *vs.* Range-Net, while n varies between $[10k - 150k]$. When $m = 50k, n = 150k$ and $r = 200$, SketchySVD has a peak memory consumption of 14GB due to oversampling parameters of $k = 801, s = 1603$, while Range-Net only requires 916MB. Since Range-Net has an exact memory requirement of $r(n + r)$ for a rank-r SVD, it always occupies main lesser memory than oversampling of SketchySVD (or any other randomized method) by two orders of magnitude, and scalable to larger datasets.

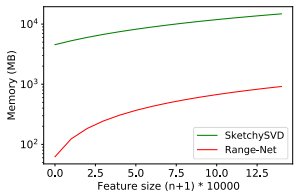


Figure 7: Peak memory load of Range-Net *vs.* SketchySVD on synthetic data.

5 DISCUSSION

From the numerical experiments it is evident that Range-Net performs at par with conventional SVD with GPU bit precision results, while being extremely memory efficient. Range-Net constructs the rank-r projector VV^T iteratively to reach the tail energy lower bound given by EYM. Alternatively, the rank-r minimizer V of **Eq. 1** gives the correct rank-r right projector VV^T of X. Any arbitrary (random) projection of X onto an oversampled rank-k subspace $(k > r)$, rather than the tail energy minimizing subspace VV^T, can inadvertently annihilate the desired top ranking singular features resulting in irreducible approximation errors.

This issue is specially important in exploratory data analysis for scientific computing, where one is not only interested in the top-most singular values of the dataset but also the dominant phenomena (singular vectors). Furthermore, with increasing digital sensor resolution the focus now is to isolate and study the lower energy (high frequency/ small spatial scale) features as in the case of Hurricane Sandy, where turbulence manifests as low rank features (see **Figs. 19, 24** in Appendix).

We again point out that accuracy is gained by achieving (or finding) this minimizer or lower bound of the rank-r tail-energy of X and therefore the upper bound on this tail-energy is of no consequence to Range-Net. As pointed out by Musco & Musco (2015), upper bounding the tail energy does not ensure that the approximation errors in the extracted singular values or vectors will reduce. Therefore, small relative tail energy errors can mislead the reader that the singular factors are equivalently accurate wherein the absolute errors can still be substantially large.

6 CONCLUSION

We present Range-Net as a low-weight, high-precision, fully interpretable neural SVD solver for big data applications that is independently verifiable without performing a full SVD. We show that our solution approach achieves lower errors metrics for the extracted singular vectors and values compared to Randomized SVD methods. A discussion is also provided on the limiting assumptions and practical consequences of using Randomized SVD schemes for big data applications. We also verify that our network minimization problems converges to the EYM tail energy bound in Frobenius norm at machine precision. A number of practical problems are considered, where SVD or Eigen decompositions are required, that demonstrate the applicability of Range-Net to large scale datasets. A fair comparison is provided against the state of the art randomized, streaming SVD algorithm with conventional SVD solution as the baseline for benchmarking and verification.

REFERENCES

Gephi sample data sets. http://wiki.gephi.org/index.php/Datasets.

Lev muchnik's data sets web page. http://www.levmuchnik.net/Content/Networks/NetworkData.html.

sklearn.utils.extmath.randomized_svd. https://scikit-learn.org/stable/modules/generated/sklearn.utils.extmath.randomized_svd.html.

James Baglama and Lothar Reichel. Augmented implicitly restarted lanczos bidiagonalization methods. *SIAM Journal on Scientific Computing*, 27(1):19–42, 2005.

Christos Boutsidis, Petros Drineas, and Malik Magdon-Ismail. Near-optimal column-based matrix reconstruction. *SIAM Journal on Computing*, 43(2):687–717, 2014.

Matthew Brand. Incremental singular value decomposition of uncertain data with missing values. In *European Conference on Computer Vision*, pp. 707–720. Springer, 2002.

François Chollet. keras. https://github.com/fchollet/keras, 2015.

Kenneth L Clarkson and David P Woodruff. Numerical linear algebra in the streaming model. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pp. 205–214, 2009.

Julio Cesar Stacchini de Souza, Tatiana Mariano Lessa Assis, and Bikash Chandra Pal. Data compression in smart distribution systems via singular value decomposition. *IEEE Transactions on Smart Grid*, 8(1):275–284, 2015.

Petros Drineas, Alan Frieze, Ravi Kannan, Santosh Vempala, and V Vinay. Clustering large graphs via the singular value decomposition. *Machine learning*, 56(1-3):9–33, 2004.

Petros Drineas, Ravi Kannan, and Michael W Mahoney. Fast monte carlo algorithms for matrices ii: Computing a low-rank approximation to a matrix. *SIAM Journal on computing*, 36(1):158–183, 2006.

Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

Gene H Golub and Richard Underwood. The block lanczos method for computing eigenvalues. In *Mathematical software*, pp. 361–377. Elsevier, 1977.

Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2): 217–288, 2011.

Liping Jing, Chenyang Shen, Liu Yang, Jian Yu, and Michael K Ng. Multi-label classification by semi-supervised singular value decomposition. *IEEE Transactions on Image Processing*, 26(10): 4612–4625, 2017.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

N Kishore Kumar and Jan Schneider. Literature survey on low rank approximation of matrices. *Linear and Multilinear Algebra*, 65(11):2212–2244, 2017.

Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

Leon Mirsky. Symmetric gauge functions and unitarily invariant norms. *The quarterly journal of mathematics*, 11(1):50–59, 1960.

Cameron Musco and Christopher Musco. Stronger approximate singular value decomposition via the block lanczos and power methods. *arXiv preprint arXiv:1504.05477*, 16:27, 2015.

Lloyd N Trefethen and David Bau III. *Numerical linear algebra*, volume 50. Siam, 1997.

Joel A Tropp, Alp Yurtsever, Madeleine Udell, and Volkan Cevher. Fixed-rank approximation of a positive-semidefinite matrix from streaming data. In *Advances in Neural Information Processing Systems*, pp. 1225–1234, 2017a.

Joel A Tropp, Alp Yurtsever, Madeleine Udell, and Volkan Cevher. Practical sketching algorithms for low-rank matrix approximation. *SIAM Journal on Matrix Analysis and Applications*, 38(4): 1454–1485, 2017b.

Joel A Tropp, Alp Yurtsever, Madeleine Udell, and Volkan Cevher. Streaming low-rank matrix approximation with an application to scientific simulation. *SIAM Journal on Scientific Computing*, 41(4):A2430–A2463, 2019.

Jalaj Upadhyay. Fast and space-optimal low-rank factorization in the streaming model with application in differential privacy. *arXiv preprint arXiv:1604.01429*, 2016.

Shuqin Wang, Yongli Wang, Yongyong Chen, Peng Pan, Zhipeng Sun, and Guoping He. Robust pca using matrix factorization for background/foreground separation. *IEEE Access*, 6:18945–18953, 2018.

Haishan Ye, Shusen Wang, Zhihua Zhang, and Tong Zhang. Fast generalized matrix regression with applications in machine learning. *arXiv preprint arXiv:1912.12008*, 2019.

Sheng Zhang, Weihong Wang, James Ford, Fillia Makedon, and Justin Pearlman. Using singular value decomposition approximation for collaborative filtering. In *Seventh IEEE International Conference on E-Commerce Technology (CEC'05)*, pp. 257–264. IEEE, 2005.