k-NN Sampling for Visualization of Dynamic data using LION-tSNE

Bheekya Dharamsotu*, K. Swarupa Rani†, Salman Abdul Moiz‡ and C. Raghavendra Rao§
School of Computer and Information Sciences
University of Hyderabad
Hyderabad, India
Email: *15mcpc21@uohyd.ac.in, †kswarupaprasad@gmail.com, ‡salman.abdul.moiz@gmail.com, §crrcs.ailab@gmail.com

***Abstract*—Dimensionality reduction algorithms are often used to visualize multi-dimensional data, which are mostly non-parametric. Non-parametric methods do not provide any explicit intuition for adding new data points into an existing environment which limits the applicability of visualization for Big Data scenario. The LION-tSNE (Local Interpolation with Outlier coNtrol t-Distributed Stochastic Neighbor Embedding) method was proposed to overcome the limitations of existing techniques. The LION-tSNE algorithm uses random sampling method for tSNE model design which creates an initial visual environment then new data points are added to this environment using local-IDW(Inverse Distance Weighting) interpolation method. The randomly selected sample data often suffer from non-representativeness of the whole data which creates inconsistency in the tSNE environment. To overcome this problem two new sampling methods are proposed which are based on k-NN (k-Nearest Neighbor) graph update properties. It is empirically shown that proposed methods outperform existing LION-tSNE method with 0.5 to 2% more k-NN accuracy and results are more consistent. The study is done on five differently characterized datasets with three different initial solutions of tSNE. The proposed method results are statistically significant which is done by statistical method pairwise t-test.**

***Index Terms*—dimensionality reduction, visualization, Big Data, t-Distributed Stochastic Neighbor Embedding(tSNE), interpolation, sampling, k-NN graph**

I. Introduction

Exploratory analysis of multi-dimensional data is a problem that arises in many areas of science such as machine learning, data mining, visualization, etc. Visualization plays a paramount role in the exploratory analysis of multi-dimensional data. The visual representation of multi-dimensional data provides an intuitive interface for an analyst to detect the underlying structure such as homogeneous regions, clusters and outliers of complex data. In visualization, the extraction of the underlying structure of multi-dimensional data is completely relying on the cognitive capabilities of the data analyst. A data analyst can percept only either 2D or 3D visual representation of the data. Obtaining 2D visual representation of multi-dimensional data is not a simple task. From the past several years, researchers have proposed various techniques to reduce the dimensionality and to visualize the multi-dimensional data. The Dimensionality Reduction (DR) [1], [2], [3] for visualization provides an opportunity to the decision makers or analysts to visualize the high-dimensional data in a low-dimensional space which provides flexibility to make effective decisions. The developed DR techniques differ with the type of structure preserved by them. The linear DR techniques such as Principal Component Analysis (PCA) [4], Factor Analysis (FA) [5], and Multi-Dimensional Scaling (MDS) [6] are used to keep two dissimilar data points faraway from each other in low-dimensional space. These techniques are incapable of reducing dimension to low-dimensional space due to the non-linearity in the multidimensional data. In contrast to linear DR, Non-linear DR techniques such as Isomap [7], Sammon mapping [8], Curvilinear Component Analysis (CCA) [9], Local Linear Embedding (LLE) [10], Laplacian Eigenmap (LE) [11], Maximum Variance Unfolding (MVU) [12] and Stochastic Neighbor Embedding (SNE) [13] are proposed to handle non-linear data. In both the approaches feature extraction depends on the characteristics of interest in the data: inter-point distances, variation, linear subspace, geodesic distances, reconstruction weights, linear tangent space, neighborhood graph and conditional probability distribution.

Student t-Distributed Stochastic Neighbor Embedding (tSNE) [14] is a popular DR technique for visualization. The main strength of tSNE is to preserve the underlying structure of multi-dimensional data in 2D or 3D space by preserving the neighborhood property of high-dimensional data in low-dimensional space. Hence, the homogeneous regions and clusters of high-dimensional data should be visible in the 2D or 3D space of tSNE. Over the last decade, tSNE has been successfully applied to visualize many applications such as genomic data [15], healthcare informatics [16], tumor subpopulations [17], etc.

Apart from classical technique Principal Component Analysis (PCA) and Self-Organizing Map (SOM) [18], most of the DR are non-parametric techniques. The characteristics of these techniques includes:(i) to only provide the projection of given data samples. (ii) do not provide any precise intuition for adding new data samples into the existing visualization, and (iii) very flexible and computationally fast. Hence, non-parametric techniques are not suitable for visualizing the time series and streaming data. Non-parametric techniques does not allow the visualization of parts of a given data and then scalable to Big Data sets on demand. Most of the existing techniques require at least quadratic computational complexity with respect to data set size which causes suffering from

2640-0316/19/$31.00 ©2019 IEEE
DOI 10.1109/HiPC.2019.00019

scalability issues.

Over the last years, researchers attempted to extend tSNE by adding new data [19], [20], [21] into existing tSNE representation. In this paper, we have proposed a new sampling technique called k-NN Sampling to extend the Local Interpolation with Outlier coNtrol tSNE (LION-tSNE) [22] technique. The LION-tSNE is a novel approach for handling the addition of new data samples in two ways. Firstly, the Inverse Distance Weight Interpolation (i.e., IDW-Interpolation) algorithm [22] is used to include the inlier data into the trained tSNE model based on local neighborhood. The local neighborhood is obtained from distance radius which is derived from the tSNE model. And the second one is Outlier_Placement algorithm [22] which used to compute the free location for placing the outliers and provides the heuristic to maintain the minimum distance between new and existing outliers. While adding new data points in a tSNE environment, the identified outliers which does not have any relation with outliers of tSNE are placed randomly in a predefined location. In LION-tSNE model, the random sampling method used for selecting a training sample from the whole population with a given fixed sample size(i.e., two-thirds of the given data). Random sampling method suffers from inefficient representativeness of the complete data that causes lots of variation in result from one execution to another, which refers to the inconsistency in the results. In this paper, we are addressing the problem of non-representativeness of the random sampling method.

The organization of the paper is as follows. In section II, we are discussing the preliminaries which provide a requirement for understanding the basics of this paper. Section III describes the related work. In section IV, we are giving a detailed description of the proposed two k-NN sampling approaches. In section V, we are providing the result analysis, evaluation, and comparison between proposed two k-NN sampling method results and the state-of-the-art technique LION-tSNE. Finally, section VI gives the direction for future work and conclusion.

II. Preliminary

A. Student t- Distributed Stochastic Neighbor Embedding (tSNE)

The tSNE algorithm developed by Laurens van der Maaten and Geoffrey Hinton [14] in 2008, based on the SNE [13] algorithm. The main idea of SNE algorithm is to represent multi-dimensional data into a human perceptual low-dimensional space. The low-dimensional representation of SNE should preserve the underlying structure of high-dimensional data in low-dimensional space by maintaining a similar neighborhood property of original data. Initially, the tSNE algorithm converts the distance between the pair of points into a conditional probability distribution. The probability distribution measures the similarity between a pair of points. The divergence between the similarities of high-dimensional data and low-dimensional embedding is measured by Kullback-Leibler divergence (i.e., KL-divergence). The minimization of the KL-divergence is obtained by the simple gradient descent that would be an objective of an SNE algorithm. The workflow of tSNE is shown in the *figure 1*.

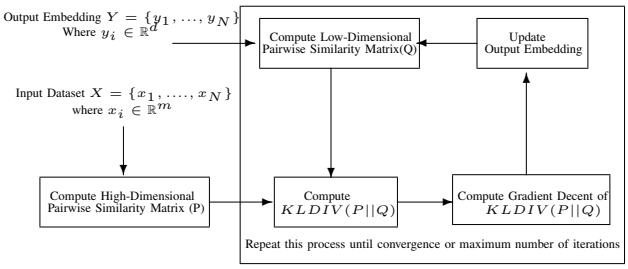


Figure 1: The tSNE workflow model

Lets assume we are given an input dataset $\mathcal{X} = \{x_1, x_2, \ldots x_N\}$ where each $x_i \in \mathbb{R}^D$ is a D-dimensional vector. The given input data can be transformed to $\mathcal{Y} = \{y_1, y_2 \ldots y_N\}$ where each $y_i \in \mathbb{R}^d$ is d-dimensional vector, $d \ll D$. The function $d(x_i, x_j)$ measures the distance between pair of points. The similarity between pair of input points x_i and x_j is denoted by $p_{j/i}$. The value of $p_{j/i}$ is the probability of choosing x_j as neighbor of x_i. The neighbors of x_i were chosen in proportion to their probability density under a Gaussian distribution with center x_i. For instance, $p_{j/i}$ is relatively high for nearby data points and infinitesimal for faraway points. The $p_{j/i}$ and symmetric p_{ij} is defined by

$$p_{j/i} = \frac{exp\left(\frac{-d(x_i,x_j)^2}{2{\sigma_i}^2}\right)}{\sum_{k\neq i}^{N} exp\left(\frac{-d(x_i,x_k)^2}{2{\sigma_i}^2}\right)},\ p_{i/i} = 0,\ p_{ij} = \frac{p_{j/i} + p_{i/j}}{2N} \quad (1)$$

Where the σ_i is the variance which can be obtained by binary search using perplexity (μ) as a given parameter. The perplexity is a smooth measure of an effective number of neighbors for each data point. The density of the data is different across Euclidean space, therefore we need different σ for each input x_i. The probability distribution P_i of x_i is obtained based on the σ_i value. The tSNE visual representation completely depends on the given parameter perplexity. The perplexity μ is obtained by Shannon entropy of P_i which gives the information gain in Information Retrieval System (IRS). When there is an increment in σ_i, then there is a relative increment in entropy. The μ and entropy is given by

$$\mu = 2^{H(P_i)} \quad where \quad H(P_i) = -\sum_{j}^{N} p_{j/i} \log_2 p_{j/i} \quad (2)$$

In low-dimensional embedding space, map-points y_i and y_j are the representative of corresponding multi-dimensional data points x_i and x_j, the similarity between y_i and y_j is denoted by q_{ij}. The q_{ij} is defined by

$$q_{ij} = \frac{(1 + d(y_i, y_j)^2)^{-1}}{\sum_{l}^{N}\sum_{k\neq l}^{N}(1 + d(y_l, y_k)^2)^{-1}},\quad q_{ii} = 0 \quad (3)$$

Here student t-distribution is used instead of Gaussian distribution. Gaussian distribution of low-dimensional embedding suffers with crowding problem [23].

If $p_{ij} \sim q_{ij}, \forall_{i \neq j} \in N$, then the given data is efficiently embedded into the low-dimensional space. Otherwise, compute the KL-divergence (i.e., error) between p_{ij} and q_{ij} that is equal to the cross-entropy in IRS. The cost function ($\mathbf{C}$) or objective of tSNE is given by

$$\mathbf{C} = KL(P \parallel Q) = \sum_{j}^{N} p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{4}$$

The cost function $\mathbf{C}$ is optimized by simple gradient descent method which is given by

$$\frac{\partial \mathbf{C}}{\partial y_i} = 4 \sum_{j} (p_{ij} - q_{ij}) q_{ij} Z (y_i - y_j) \tag{5}$$

where $Z = (1 + d(y_i, y_j)^2)^{-1}$ is a normalization term of student t-distribution. The updatation of map-points are obtained by applying the *equation 5* which computes the attractive and repulsive forces applied among map points. If two map points are similar, then the attractive force applied; otherwise, the repulsive force applied. The optimization of cost function $\mathbf{C}$ is suffering from local minima problem which can be resolved by applying early exaggeration [14] to input similarity. After several simple gradient descent iterations, the tSNE model converges the underlying structure of the input data.

B. Sampling Clustering (SC)

Sampling Clustering (SC) [28] constructs a lite informative non-binary dendrogram by recursively dividing a graph into subgraphs until there are no vertices in the graph. In each recursive call, the graph nodes are sampled to disconnect latent sub-clusters. The sampled nodes are removed from the graph along with edges. After each sampling, condense is applied to avoid the fragmentation of clusters; this allows the addition of new edges to remaining nodes by connecting the other nearest neighbor points of a graph. The process of sampling is shown in *figure 2*. This work motivated us to propose new sampling methods.

C. Inverse Distance Weight Interpolation (IDW-Interpolation)

IDW-Interpolation [26] map new data point $x \in \mathcal{R}^D$ into the existing embedding $y_i \in \mathcal{R}^d$, where $\{i = 1, 2, ...m\}$ and $m < N$. IDW-Interpolation determine the value of x as weighted sum of values y_i, where weight is proportional to inverse distances. The IDW-Interpolation of x is given by

$$IDWI(x) = \sum_{\|x - x_i\| \leq r} w_i(x).y_i,$$
$$Where \;\; w_i(x) = \frac{\parallel x - x_i \parallel^{-p}}{\sum_{\|x - x_i\| \leq r} \parallel x - x_i \parallel^{-p}} \tag{6}$$

For instance, when the data point $x \rightarrow x_i$, the inverse distance $\parallel x - x_i \parallel^{-1} \rightarrow \infty$, the corresponding weight $w_i(x) \rightarrow 1$ (i.e., $\forall_{j \neq i} w_j(x) \rightarrow 0$ due to the normalization) and $IDWI(x) \rightarrow y_i$. In our expermentation we use local IDW-Interpolation which consider only limited number of neighbor points for computing the value of x. The neighbor points selection is done by given parameter radius r_{xNN}. The parameter r_{xNN} value is calculated by the heuristic proposed by Andrey Boytsov et.al. [22]. In local IDW-Interpolation, the power parameter p plays an important role. For instance, very small value of p predict the value of x around the center: $y \approx mean(y_i)$ (unless $x = x_i$) even the distance $\parallel x - x_i \parallel$ is low because the weight distribution is close to uniform. When the power parameter is high and the distance $\parallel x - x_i \parallel$ is low, the weight $w_i(x)$ of very first nearest neighbor is dominating all other neighbors, therefore $y \approx y_i$ where $i = argmin \parallel x - x_j \parallel$. Too small and too large values of p suffers with different forms of overfitting. In LOIN-tSNE [22], authors proposed a generalization of power parameter by using leave-one-out cross-validation which is applied on training sample. Based on leave-one-out cross validation they computed the local IDW-Interpolation for each training sample that produce the estimation of each y_i. Afterwards, the mean square distance between estimated y_i's and real y_i's is computed. While optimizing power parameter the mean square error should be minimum. The obtained power parameter considered as a metric. However, this metric is heuristic, not an exact criteria.

D. (t, m, s)-Nets and (t, s)-Sequences

The (t, m, s)-Nets and (t, s)-sequences [24] combines the Jittered and Latin Hypercube sampling in order to achieve more uniformity and general concepts of stratification (*see figure 3(b)*). Jittered sampling divides the data space into N equal sized cubes then select one sample from each cube randomly, where $N = n \times m$ and $n \sim m$ in 2D coordinated dataset(*see figure 3(a)*). In Latin hypercube sampling each data coordinate is divided into N equal intervals. Then the samples are chosen randomly such that each interval contains exactly one point (*see figure 3(c)*). For instance, 16-sample 2D representation of Jittered, Latin Hypercube and $(0, 4, 2)$-Nets samplings is shown in *figure 3*. In *figure 3*, (t, m, s)-Nets impose more stratification for uniform distribution across the given population. Therefore, the representativeness of the population is maintained more uniformly.

III. Related Work

To incorporate new data into the existing tSNE environment, most of the existing technique designed a mapping function $f : \mathcal{X} \rightarrow \mathcal{Y}$, which accepts multi-dimensional data and returns its low-dimensional embedding. Obtained mapping functions are used for incorporating the new data points into the tSNE environment.

Laurens van der Matten [19], the author of the original tSNE algorithm, has proposed parametric tSNE that learn both tSNE representation and mapping together. The mapping function of parametric tSNE is obtained by neural network concept Restricted Boltzmann Machines(RBM) [29] which is used for building an approximation of tSNE mapping. Gisbrecht et.al. [20] proposed Kernel-tSNE which has close relation to RBF-Interpolation [25] and IDW-Interpolation [26]. Kernel-tSNE is a parametric tSNE designed based on the normalized Gaussian kernels.

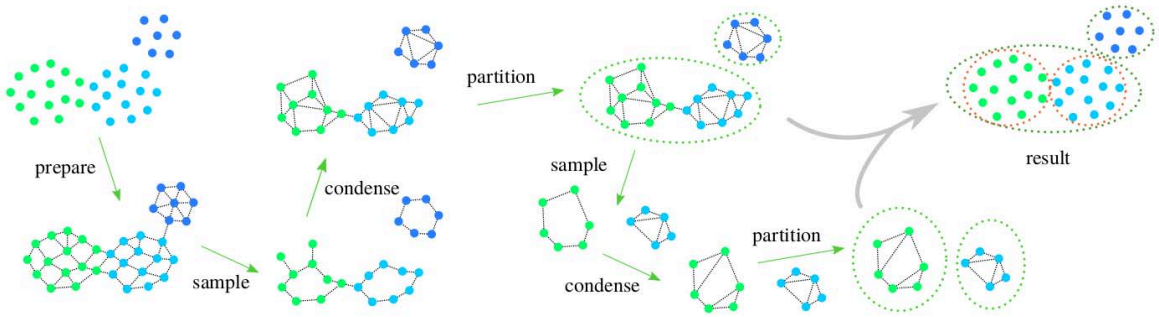


Figure 2: An ideal flow diagram of sampling cluster. First the dataset is converted into a graph. The graph is sampled, condensed and partitioned into small graphs.The same operations are performed until the termination condition is reached.

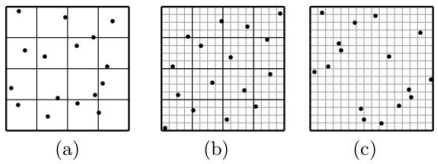


Figure 3: Realization of *(a) Jittered sampling (b) (0,4,2)-Nets and (c) Latin Hypercube sampling*

Pezotti et.al. [21] proposed Approximated-tSNE for Progressive Visual Analytics. This algorithm approximates the distance function using k-Nearest Neighbors for tSNE embedding and also used the degree of approximation to show the trade-off between embedding quality and computation speed up.

Andrey Boytsov et.al. [22] proposed LION-tSNE algorithm based on local IDW-Interpolation for adding new data sample into an existing tSNE environment. It also addresses the outlier handling approaches. In this paper, we are extending the idea of LION-tSNE algorithm by proposing k-NN sampling method for the selection of training samples. It allows the samples selection with respect to their k-nearest neighbors instead of random sampling for tSNE model design. For more efficient sampling in the context of visualization review [34] which selects sample according to the specified group constraint preservation. In our proposed approach, we are choosing samples according to their preference which is calculated by neighborhood property of each data points. In our paper, we are addressing the lack of representativeness problem which is caused by random sampling in LION-tSNE. Therefore, in our experimental evaluation, we are comparing proposed methods results with only LION-tSNE. For detailed comparative study, refer Andrey Boytsov et.al. [22] paper.

IV. Proposed method

The framework of the proposed method is shown in the *figure 4*. It has four stages, at stage 1 data preprocessing will be done such as the removal of redundant data points and filling the empty variables with appropriate values. At stage 2, the proposed k-NN sampling is done for selecting train_sample which is described in section *A*. At stage 3, the selected train_samples projected into a low-dimensional embedding with Barnes-Hut tSNE (BH-tSNE) [27] and new data points are added into the tSNE model that discussed in section *B*. At Last k-NN accuracy is calculated for whole data as discussed in section *C*.

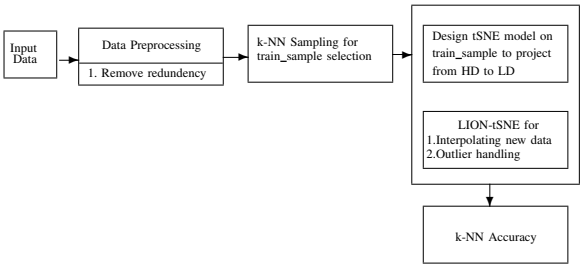


Figure 4: Proposed framework of k-NN sampling for visualization of dynamic data using LION-tSNE

A. k- Nearest Neighbor sampling (k-NN sampling)

In our proposed k-NN sampling method, we are computing Nearest Neighbor score (NN_score) and Mutual Nearest Neighbor score (MNN_score) from the k-NN graph. Lets assume k-NN graph is a directed graph $G = (V, E)$, the edge $E(v_1, v_2)$ gives v_2 as a neighbor of v_1 and neighborhood of v_1 is denoted by N_{v1}. The *out-degree* of each vertex is equal to k, and the *in-degree* of a vertex depends on the neighborhood property of other vertices. In our method, each data point is considered as a vertex of the k-NN graph and k is taken as a parameter for selecting the optimal training sample. The NN_score of data point x_i is equal to the *in-degree* of x_i which is given by

$$NN\_score(x_i) =\mid \{x_j \mid x_i \in N_{x_j}\} \mid, \forall_{j \neq i} x_j \in \mathcal{X} \tag{7}$$

where $\mathcal{X}$ denotes whole data set, N_{x_j} denotes the neighborhood of x_j. The MNN_score of data point x_i is at most k which is given by

$$MNN\_score(x_i) =\mid \{x_j \mid x_i \in N_{x_j} \wedge x_i \mid x_j \in N_{x_i}\} \mid, \forall_{j \neq i} x_j \in \mathcal{X} \tag{8}$$

The data point x_i is selected as a train_sample and which can have representativeness of its neighbors is given by

$$train\_sample = first\_index\{argmax_{x_i \in \mathcal{X}} \{NN\_score(x_i)\} \cap argmax_{x_i \in \mathcal{X}} \{MNN\_score(x_i)\}\} \quad (9)$$

Proposed k-NN sampling algorithm shown in *algorithm 1*. Initially, the train_sample is a null set; at each iteration, one data point is appended to the train_sample. At each iteration, train_sample as well as its mutual neighbors, are deleted from the $\mathcal{X}$ and then k-NN graph is updated. For k-NN graph updation we are proposing two different strategies, one is static k-NN graph updation and another one is dynamic k-NN graph updation. The train_samples are selected according to the above two methods. In both approaches, the train_sample selection is repeating until the NN_score of remaining $\mathcal{X}$ is equal to zero or size of remaining $\mathcal{X}$ is less than k of k-NN graph. Later, if the size of the remaining $\mathcal{X}$ is equal to zero, then the selected train_sample represents the complete data, and it is considered to be good. Otherwise, the data points are sampled from the remaining $\mathcal{X}$ using (t,m,s)-Nets. The samples which are selected using (t,m,s)-Nets are appended to the train_sample for improving the representativeness of the data. The computational complexity of proposed method is $O(N^2)$ which limits the algorithm scalable to large datasets. To get the effective insights from the obtained NN_score's and MNN_score's we would like to visualize the k-NN graphs of proposed methods with terrain metaphor [33] which can be done in future work. The embedding of selected train_sample and addition of new data into an existing tSNE environment is described in the following section.

Algorithm 1: k-NN Sampling algorithm

Data: data set $\mathcal{X} = \{x_1, x_2, ...x_N\}$, parameter k for minimal train_sample
Result: Return train_sample
$train\_sample = \emptyset$
begin
  Compute k-NN graph of $\mathcal{X}$
  repeat
    Compute $NN\_score(\mathcal{X})$
    Compute $MNN\_score(\mathcal{X})$
    $index = [NN\_score(\mathcal{X}) == argmax\{NN\_score(\mathcal{X})\}]$ /* gives all index which are having same NN-score */
    if $len(index) > 1$ **then**
      $train\_index = argmax\{MNN\_score(x_i)\}$ where $i \in index$
    end
    else
      $train\_index = index$
    end
    $train\_sample = train\_sample \cup \mathcal{X}[index]$
    Determine the mutual neighbors of $index$
    Delete the selected $index$ as well as its mutual neighbors from $\mathcal{X}$
    if *sampling is dynamic* **then**
      Re-compute k-NN graph of remain $\mathcal{X}$
    end
  until $(NN - score(\mathcal{X}) = 0) \vee (\mid train\_sample \mid \leq k)$;
  if $\mathcal{X}$ *is not Null* **then**
    $train\_sample = train\_sample \cup (t, m, s) - Nets(\mathcal{X})$
  end
end

B. Low-dimensional embedding of data

1) tSNE model: train_sample embedding: Barnes-Hut tSNE (BH-tSNE) [27] algorithm is used to generate low-dimensional embedding of selected train_sample data. BH-tSNE is an optimized version of tSNE; it optimizes the tSNE objective function by input similarity approximation and gradient descent approximation. The original BH-tSNE algorithm takes initial embedding points randomly and then applies early exaggeration on original data. The process of early exaggeration is used to move two similar points near to each other in embedding space. The random initial solution takes more number of iteration for convergence. To overcome this problem, in our experimentation, we are considering two more initial solutions such as PCA and MDS for better accuracy and to reduce the divergence cost. For adding new data points into a tSNE model is discussing in the below section.

2) LION-tSNE: Interpolation and Outlier handling: The new data points are added to the tSNE model in two ways. The new data point addition will be done according to the calculated parameter, r_{xNN}, r_{yNN} and r_{close}. The parameter r_{xNN} is the minimum radius of input data that decides whether the given new data point is either inlier or outlier based on the heuristic proposed by Andrey Boytsov et.al [22]. These parameters are derived from the designed tSNE model. Inlier data points are interpolated into the designed tSNE model using local IDW-Interpolation method. The identified outliers which are no more related to outliers of tSNE they can be placed randomly into the predefined locations. In otherwords, the identified outliers have some relation with outliers of tSNE that are placed near to them by using parameter r_{close}. The parameter r_{yNN} is the minimum distance from data points to outliers and from outlier to outlier, and the outlier locations are determined by the outlier_placement algorithm using r_{yNN} based on the heuristic proposed by Andrey Boytsov et.al [22].

C. k-NN accuracy

The newly added data samples should be embedded among the training samples of similar characteristics. To evaluate this, for each newly added data sample k-NN accuracy is calculated in embedding space, and observe the percentage of k neighbors which are having the similar characteristics. The k-NN accuracy typically depends on the parameter k value which is considered as fixed in our experimentation. For instance, the smaller value of k will give good performance accuracy and while increasing the k value performance accuracy will decrease. The selection of parameter k is also playing a paramount role in k-NN accuracy measure. The relationship between k-value and accuracy is shown in the *figure 5*.

V. Evaluation

To evaluate the proposed method, The results of proposed methods are compared with the state-of-the-art technique

LION-tSNE. We performed experiments on five numerical datasets that represent applications of various domains.

The section *A* describes the datasets which are used in our experiments. The experimental configuration explored in section *B*. In section *C*, we present the experimental evaluation among the results of our proposed methods based LION-tSNE and random sampling based LION-tSNE. The evaluation is done by considering three different initial solutions such as PCA, MDS and random for designing initial tSNE model.

A. Datasets

In our experimental evaluation, we have considered five differently characterized datasets such as IRIS dataset, Breast-Cancer dataset, Wine dataset, MNIST dataset, and Olivetti-Faces dataset. In Olivetti-Faces dataset, ten images of one individual were considered as small variation in viewpoint, large variation in expression, and addition of glasses. The table *I* provides the detailed description of all five datasets.

Dataset Name	Size	Dimensions	Classes
IRIS	150	4	3
Breast Cancer	569	30	2
Wine	178	13	3
MNSIT	70K	784	10
Olivetti faces	400	10304	40

Table I: Overview of datasets along with their size, dimensions, and classes

B. Experimental configuration

In all of our experimental evaluation, we have used three different initial solutions for embedding of BH-tSNE that are PCA, MDS, and random. The PCA and MDS initial solutions reduce the divergence cost. The datasets with more than 30 dimensions, their dimensionality is reduced to 30 dimensions by PCA. This process speeds up the computation of the probability distribution of the input similarity and suppresses some noise. The results of BH-tSNE algorithm are shown in scatter-plot representation. For all the datasets, there is a class information of each data point which can be used only for visual clarity of a scatter-plot. The data point class information is not used for any other purposes.

Due to the computational complexity of proposed method and BH-tSNE algorithm (i.e., computational complexity $O(NlogN)$) we have considered limited dataset size for computation. For our experimentation, we have considered the maximum dataset size as 10K. There is a scope for expansion of dataset size by extending the implementation in distributed environment.

Table *II* provides the parameter settings of our experimental evaluation. The parameter *perplexity* represents the effective number of neighbors for each data point. For instance, the small value of *perplexity* creates subgroups within the same cluster of tSNE results. In contrast to small, the large value of it does not maintain the clear separation between two clusters of tSNE results. In both cases, there is a lack of visual clarity, the empirical studies state that the *perplexity* value between 5 to 70 gives a good visual representation of tSNE results. The parameter *dist_per* represents the overall percentage of train_sample points considered as inliers. For example, if we take *dist_per* as 95th percentile that means out of 100 points, 95 points are considered as inliers and remaining 5 points are considered as an outlier. The parameter k plays an important role in the computation of k-NN accuracy of the data. The effect of parameter k is shown in the *figure 5*. It is clear that when there is an increment in k value, accuracy decrease monotonically. The *figure 5* shows the k-NN accuracy of including and excluding outliers. Because of the paper limitations, we are evaluating the proposed k-NN sampling method results using only one dataset (i.e., MNIST dataset of size 10K).

Parameter	Value
Perplexity	5 - 30
r_{xNN} at *dist_perc*	100
Fixed k-value for k-NN accuracy	10, 3(for Olivetti-face dataset)

Table II: Parameters setting for the experimental setup

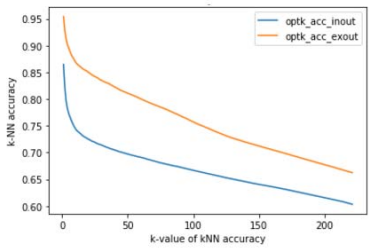
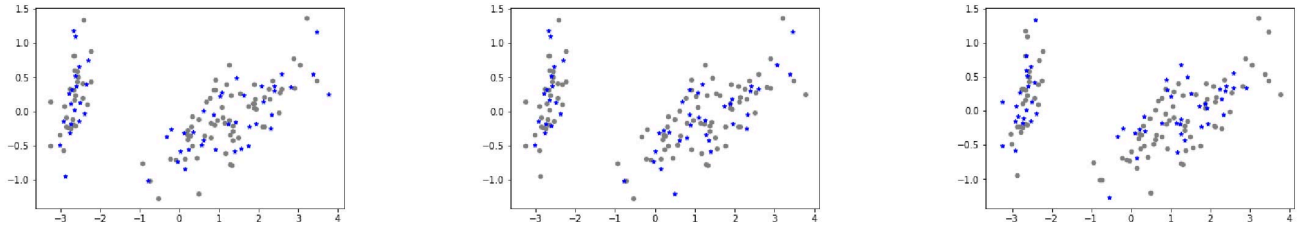


Figure 5: Relationship between k-value of k-NN accuracy and k-NN accuracy which is defined on optimal train_sample size.

C. Results

In *figure 6*, we show the experimental train_sample selection results of proposed k-NN sampling methods(with static and dynamic k-NN graph update) and random sampling on IRIS dataset. For clear 2D visual clarity, we have chosen IRIS dataset because original dimensionality of the IRIS dataset is four which is very small. So, the direct 2D visual representation of 4D data (i.e., IRIS data) does not show the huge variation in scatter-plot representation. The train_sample selection (i.e., represented by blue star scatter-plot point) results in sub-figure $6(a)$ and $6(b)$ of the proposed k-NN sampling methods clearly shows the unified sampling of complete data. In sub-figure $6(c)$, the results of random sampling shows that some regions of the data are not covered by selected sample and random sampling is not consistent. Due to the inconsistency of random sampling, there is a lack of representativeness which causes the inefficient result of tSNE.

The relationship between train_sample selection and fixed k-NN accuracy is shown in the *figure 7*. The figure $7(a)$ and $7(b)$ shows the realization of k-NN accuracy with including and excluding outliers and train_sample_size. In both the figures, it is clear that the k-NN accuracy of both including and



(a) Static k-NN sampling (b) Dynamic k-NN sampling (c) Random sampling

Figure 6: Training sample selection (i.e., represented by blue star scatter-plot point) from IRIS dataset using proposed k-NN sampling methods(with static and dynamic k-NN graph update) and random sampling

excluding outliers is monotonically reducing while increasing the k-value of k-NN sampling. The corresponding random sampling results also realized beside proposed sampling results (i.e, titled as Random sampling). There is a small fluctuation in the accuracy curve which is caused by additional train_sample selection from the remaining list using (t,m,s)-Nets. The train_sample_size is also decreasing while increasing the k-value of k-NN sampling. The relationship between k-NN accuracy of including and excluding outliers is defined by the Pearson correlation coefficient [30]. For all datasets, the correlation coefficient is approximately equal to one which means both including and excluding outliers k-NN accuracies are more similar to each other (that means optimal solution has very small number of outliers). The relationship between train_sample_size and fixed k-NN accuracy is derived by linear regression model [31] which is shown in the figure $7(c), 7(d)$ and $7(e)$. The linear regression model for three different situations will produce good regression score (i.e., above 0.9). Empirically, the regression score is more than 0.5 means the y variable (i.e., y-axis) is well defined by variable x (i.e., x-axis).

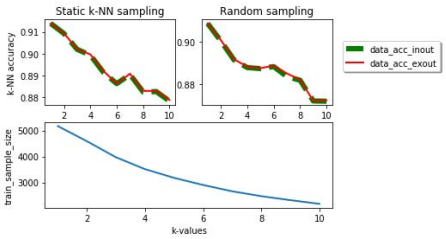
The *figure 8* gives the 2D visual representation of MNIST dataset of 10K points with PCA based initial solution. Sub-figures $8(a)$, $8(d)$ and $8(g)$ shows train_sample scatter-plots which is obtained by BH-tSNE model. Sub-figure $8(a)$ and $8(d)$ shows the scatter-plots of proposed methods; static and dynamic k-NN sampling. The figure $8(g)$ shows the scatter-plot of random sampling of a size equivalent to the static variant of the proposed method. The figures $8(b)$, $8(c)$, $8(e)$, $8(f)$, $8(h)$ and $8(i)$ shows the scatter-plots of LION-tSNE of all three methods. The figures $8(b)$, $8(e)$ and $8(h)$ gives the scatter-plots of three methods before power parameter optimization. The figures $8(c)$, $8(f)$ and $8(i)$ shows the scatter-plots of all three methods after power parameter optimization. After power optimization, the scatter-plots of the proposed methods are more clear and effective than the state-of-the-art technique.

In our result analysis, we are comparing the k-NN accuracies of proposed k-NN sampling and random sampling based LION-tSNE results. We are also using three different initial solutions such as PCA, MDS and random. The empirical study states that the results of MDS are computationally slower than the results of PCA. The PCA and MDS initial solutions reduce the computational complexity of the original tSNE model by early convergence. The accuracies of proposed k-NN and random sampling based LION-tSNE results are assessed with baseline accuracy. Baseline accuracy is obtained by applying tSNE on overall data. The baseline accuracies of five datasets are shown in the *table III*. The k-NN accuracy comparison of proposed k-NN and random sampling based LION-tSNE results on MNIST dataset with three different initial solutions is shown in the *figure 9* along with baseline accuracy. The figure $9(a)$, $9(b)$ and $9(c)$ clearly shows that the k-NN accuracies of proposed k-NN sampling based LION-tSNE outperforms existing state-of-the-art that is random sampling based LION-tSNE in all three initial solution constraint. Also, proposed methods accuracy results are almost equal to baseline results. For some datasets such as IRIS, Breast-Cancer, and Wine, the proposed methods accuracy result is more than baseline accuracy. The k-NN accuracies of proposed k-NN and random sampling based LION-tSNE results on five datasets with three different initial solutions are listed in table *IV* and *V*.

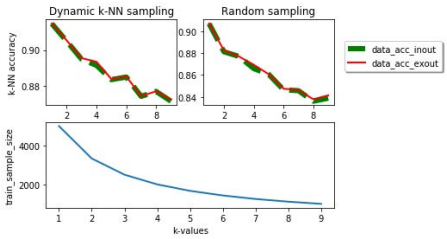
Dataset	Dataset size	Baseline Accuracy
IRIS	150	0.95238
Breast-Cancer	569	0.91282
Wine	178	0.69943
MNIST Digits	10000	0.93298999
Olivetti face	400	0.92416

Table III: k-NN accuracy of static variant of proposed method for various datasets

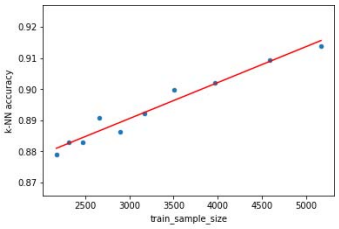
Tabel *IV* shows the static k-NN and random sampling based k-NN accuracies of LION-tSNE on all five datasets. For all five datasets with all three initial solution constraints and same train_sample size, the static k-NN sampling based LION-tSNE is giving better k-NN accuracies than random sampling based LION-tSNE. In tabel *V*, the k-NN accuracies of dynamic k-NN and random sampling based LION-tSNE are listed on all five datasets. Here also, the dynamic k-NN sampling based LION-tSNE is reporting better k-NN accuracies than random sampling based LION-tSNE in all circumstances. If we compare the results of both proposed methods which are shown in table *IV* and *V*. The static method gives more accuracy than dynamic one. We also observed that in most of the cases both static and dynamic k-NN sampling methods selecting optimal train_sample at k-value either one or two of k-NN graph.



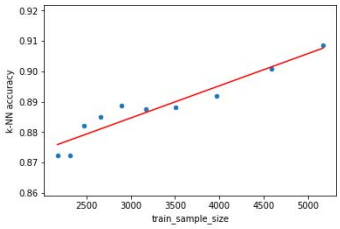
(a) Realization of train_sample_size and k-NN accuracy of static k-NN and random sampling based LION-tSNE results with ten different k-values



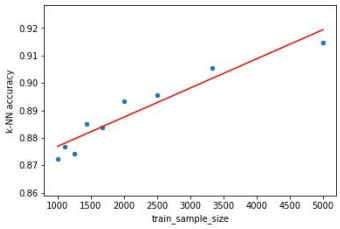
(b) Realization of train_sample_size and k-NN accuracy of dynamic k-NN and random sampling based LION-tSNE results with ten different k-values



(c) Relationship between train sample size and k-NN accuracy of static k-NN sampling method



(d) Relationship between train sample size and k-NN accuracy of random sampling method



(e) Relationship between train sample size and k-NN accuracy of dynamic k-NN sampling method

Figure 7: Realization of train sample size and fixed k-NN accuracy, relationship between train sample size and fixed k-NN accuracy with liner regression model.

Dataset	Initial solution type	Train sample size	Static k-NN sampling			Ran.sampling	
			Opt.k-value	Opt. power	k-NN accur	Opt. power	k-NN accur
IRIS	PCA	54	3	2	0.96068	1	0.94246
	MDS	44	5	2	0.96382	9	0.94701
	Random	66	2	2	0.95586	5	0.94082
Breast Cancer	PCA	108	9	46	0.91265	49	0.90530
	MDS	80	14	15	0.91569	11	0.89557
	Random	63	19	3	0.91058	32	0.90353
Wine	PCA	70	3	1	0.69943	3	0.68563
	MDS	36	7	6	0.70505	44	0.67901
	Random	50	5	2	0.71348	1	0.70568
MNIST Digits	PCA	5169	1	49	0.91387	45	0.90855
	MDS	4586	2	48	0.91882	49	0.89989
	Random	5118	1	48	0.91879	45	0.90869
Olivetti Face	PCA	217	1	31	0.91583	41	0.88719
	MDS	220	1	39	0.915	40	0.89313
	Random	221	1	40	0.92333	26	0.89983

Table IV: k-NN accuracy results of LION-tSNE for static k-NN and random sampling on five datasets

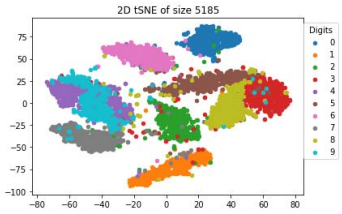
Dataset	Initial solution type	Train sample size	Dynamic k-NN sampling			Ran.sampling	
			Opt.k-value	Opt. power	k-NN accur	Opt. power	k-NN accur
IRIS	PCA	49	2	18	0.96013	1	0.95546
	MDS	49	2	6	0.96293	3	0.9531
	Random	49	2	21	0.96223	32	0.95174
Breast Cancer	PCA	241	2	26	0.90984	9	0.90598
	MDS	191	2	30	0.91232	19	0.90896
	Random	152	5	5	0.91669	43	0.90371
Wine	PCA	68	3	1	0.71404	4	0.69548
	MDS	89	1	5	0.71129	1	0.69017
	Random	69	3	1	0.70617	35	0.69550
MNIST Digits	PCA	5000	1	49	0.91592	49	0.90700
	MDS	5000	1	48	0.92032	49	0.91221
	Random	5000	1	42	0.91324	49	0.90984
Olivetti Face	PCA	200	1	41	0.91	49	0.87786
	MDS	200	1	46	0.91916	47	0.87955
	Random	200	1	23	0.92499	22	0.88416

Table V: k-NN accuracy results of LION-tSNE for static k-NN and random sampling on five datasets

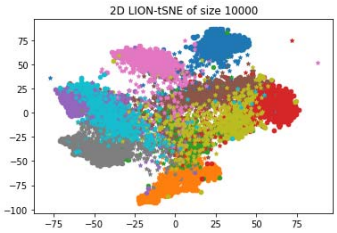
The proposed k-NN sampling methods are statistically significant than the random one which is proved by the statistical method pairwise t-test [32]. For pairwise t-test, we have repeated out experiments 25 times on all five datasets. Each time we have collected optimal LION-tSNE accuracy results of proposed methods as well as corresponding LION-tSNE results of random sampling method. The proposed k-NN sampling based LION-tSNE gives more consistent results than random one which is shown in the *figure 10*. The *figure 10(a)* and *10(b)* clearly shows the superiority of proposed k-NN sampling over random sampling. The *figure 10(c)* shows the results of static k-NN sampling are more accurate than dynamic k-NN sampling. For evaluation, we have conducted three pairwise t-test such as static k-NN vs random, dynamic k-NN vs random and static k-NN vs dynamic k-NN. From all three t-test for all five datasets, we have obtained the positive values for t-statistic and p-values are less than 0.05. The sample pairwise t-test of MNIST dataset is shown in the table *VI*. The empirical studies states that the p-value of pairwise t-test is less than 0.05 then the results are statistically significant. Therefore, the proposed k-NN sampling methods are more consistent and efficient.

t-test pair	t-statistic value	p-value
(static k-NN, random)	10.908428	4.38×10^{-11}
(dynamic k-NN, random)	5.233352	1.15×10^{-5}
(static k-NN, dynamic k-NN)	12.709989	1.88×10^{-12}

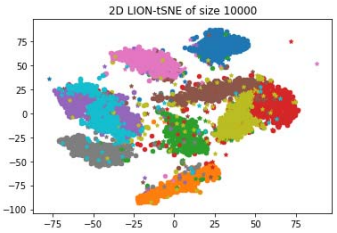
Table VI: Pairwise t-test results on MNIST dataset of size 10K for evaluation of statistical significance



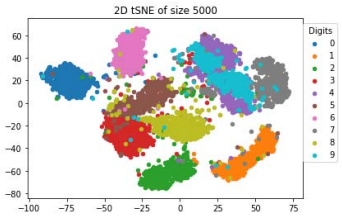
(a) Proposed static k-NN sampling based 2D visual representation of tSNE



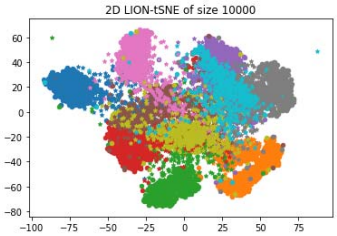
(b) Proposed static k-NN sampling based 2D visual representation of LION-tSNE before power parameter optimization



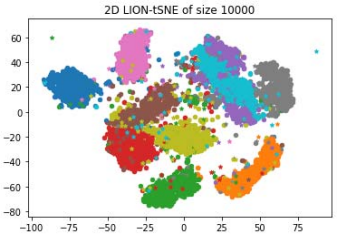
(c) Proposed static k-NN sampling based 2D visual representation of LION-tSNE after power parameter optimization



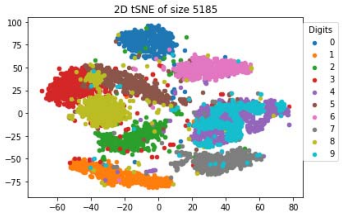
(d) Proposed dynamic k-NN sampling based 2D visual representation of tSNE



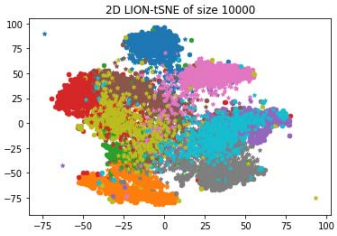
(e) Proposed dynamic k-NN sampling based 2D visual representation of LION-tSNE before power parameter optimization



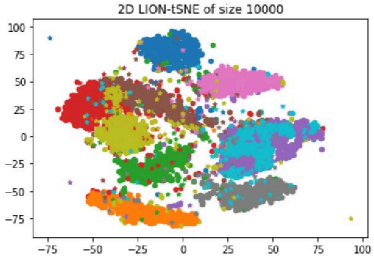
(f) Proposed dynamic k-NN sampling based 2D visual representation of LION-tSNE after power parameter optimization



(g) Random sampling based 2D visual representation of tSNE



(h) Random sampling based 2D visual representation of LION-tSNE before power parameter optimization



(i) Random sampling based 2D visual representation of LION-tSNE after power parameter optimization

Figure 8: Three different 2D visual representation of tSNE and LION-tSNE on MNIST dataset of size 10K

VI. Conclusion and Future Work

This paper deals with adding new data samples into the existing tSNE environment to overcome the scalability issues. We proposed two new approaches such as static and dynamic k-NN graph update methods (i.e., k-NN sampling) for good tSNE model design. This method selects the unified train_samples from entire data. The proposed methods give good samples to overcome the lack of representativeness. Our experiments on five datasets show that proposed methods outperform existing state-of-the-art techniques for adding new data point into an existing tSNE environment and also handled outliers.

Proposed method deals with numerical datasets which can be enhanced in future to categorical and mixed datasets. Due to the computational complexity, the proposed method is suitable to limited dataset size. To overcome this limitation, it can be implemented in distributed environment. In our work, we are assuming some of the data points are outliers from tSNE enviroment; instead of that we would like to deal outliers within the tSNE itself by introducing threshold value. Also, to get clear insights from the obtained NN_score and MNN_score, it is good to view the visual representation of k-NN graph. Therefore, we would like to use the method of Yang Zhang et.al. [33] to visualize the relationship among the data points.

References

[1] L. van der Maaten, E. Postma, and H. van den Herik. *Dimensionality reduction: A comparative review*. Technical report, Tilburg University Technical Report, TiCC-TR 2009-005, 2009.

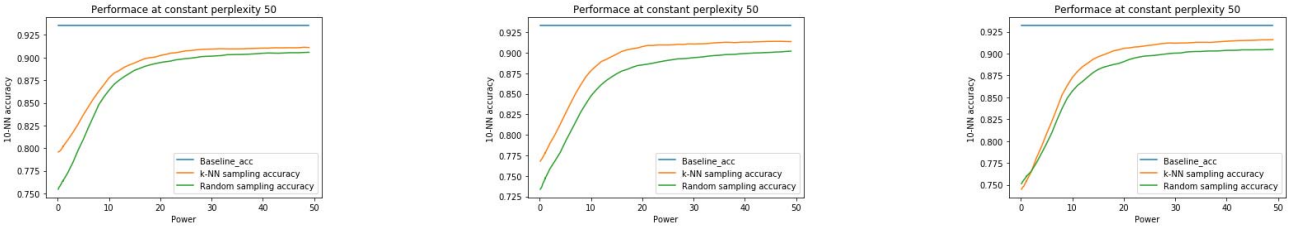
[2] J. Venna, J. Peltonen, K. Nybo, H. Aidos, and S. Kaski. *Information retrieval perspective to nonlinear dimensionality reduction for data visualization*. Journal of Machine Learning Research, 11:451-490, 2010.

[3] K. Bunte, M. Biehl, and B. Hammer. *A general framework for dimensionality reducing data visualization mapping*. Neural Computation, 24(3):771-804,2012.

[4] M. Partridge and R. Calvo. *Fast dimensionality reduction and Simple PCA*. Intelligent Data Analysis, 2(3):292-298, 1997.

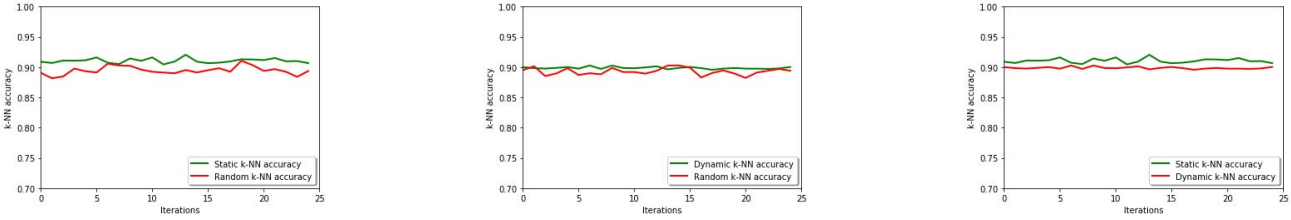
[5] J. Shi and J. Malik. *Normalized cuts and image segmentation*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(8):888-905, 2000.

[6] Trevor F. Cox,Michael A. A. Cox. *Multidimensional scaling*, 2nd edition,Chapman & Hall/CRC, 2001.



(a) MDS initial solution (b) PCA initial solution (c) Random initial solution

Figure 9: Performance evaluation of proposed static k-NN sampling based LION-tSNE accuracy results with baseline accuracy for three different initial solutions of MNIST dataset of size 10K.



(a) Static k-NN vs random sampling (b) Dynamic k-NN vs random sampling (c) Static vs dynamic k-NN sampling

Figure 10: Comparison of three different paired results of LION-tSNE over 25 intervals on MNIST dataset of size 10K. The initial solution is PCA based.

[7] J. Tenenbaum, V. da Silva, and J. Langford. *A global geometric framework for nonlinear dimensionality reduction*. Science, 290:2319-2323, 2000.

[8] J.W. Sammon. *A nonlinear mapping for data structure analysis*. IEEE Transactions on Computers, 18(5):401-409, 1969.

[9] P. Demartines, J.Herault, *Curvilinear Component Analysis: A Self-Organizing Neural Networks for nonlinear mapping of datasets*, IEEE Trans. Neural Networks, vol.8, pp.1197-1206, 1997.

[10] S. T. Roweis and L. K. Saul., *Nonlinear dimensionality reduction by locally linear embedding*, Science, 290: 2323 - 2326, 2000.

[11] M. Belkin and P. Niyogi. *Laplacian eigenmaps for dimensionality reduction and data representation*. Neural Computation, 15:1373-1396, 2003.

[12] K. Weinberger and L. K. Saul., *An introduction to nonlinear dimensionality reduction by maximum variance unfolding*. In Proceedings of the National Conference on Artificial Intelligence, pages 1683-1686, Boston, MA, 2006.

[13] Geoffrey E Hinton and Sam T. Roweis. *Stochastic Neighbor Embedding*, Advances in Neural Information Processing Systems 15, pages 857-864. MIT Press, 2003.

[14] Laurens van der Maaten and Geoffrey Hinton, *Visualizing Data using t-SNE*, Journal of Machine Learning Research, 9(Nov):2579-2605, 2008.

[15] Wentian Li, Jane E. Cerise, Yaning Yang, and Henry Han. *Application of t-SNE to human genetic data*, J. Bioinform. Comput. Biol. (2017), p. 1750017, 10.1142/S0219720017500172

[16] Minh Nguyen, Sanjay Purushotham, Hien To, and Cyrus Shahabi. *m-TSNE: A Framework for Visualizing High-Dimensional Multivariate Time Series*, In VAHC2016 Workshop on Visual Analytics in Healthcare in conjunction with AMIA 2016, 2016.

[17] Walid M. Abdelmoula, Benjamin Balluff, Sonja Englert, Jouke Dijkstra, Marcel J. T. Reinders, Axel Walch, Liam A. McDonnell, and Boudewijn P. F. Lelieveldt. *Data-driven identification of prognostic tumor subpopulations using spatially mapped t-SNE of mass spectrometry imaging data*, Proceedings of the National Academy of Sciences, 113(43):12244-12249, October 2016.

[18] T. Kohonen. *Self-organization and associative memory: 3 rd edition*. Springer-Verlag New York, Inc., New York, NY, USA, 1989.

[19] Laurens van der Maaten, *Learning a Parametric Embedding by Preserving Local Structure*, In International Conference on Artificial Intelligence and Statistics, pages 384391, 2009.

[20] Andrej Gisbrecht, Alexander Schulz, and Barbara Hammer. *Parametric nonlinear dimensionality reduction using kernel t-SNE*, Neurocomputing, 147:7182, January 2015.

[21] N. Pezzotti, B. P. F. Lelieveldt, L. v d Maaten, T. Hllt, E. Eisemann, and A. Vilanova. *Approximated and User Steerable tSNE for Progressive Visual Analytics*, IEEE Transactions on Visualization and Computer Graphics, 23(7):17391752, July 2017.

[22] Boytsov A, Fouquet F, Hartmann T, LeTraon Y. *Visualizing and exploring dynamic high-dimensional datasets with LION-tSNE*, arXiv:1708.04983, 2017.

[23] J.A. Cook, I. Sutskever, A. Mnih, and G.E. Hinton. *Visualizing similarity data with a mixture of maps*, In Proceedings of the 11 th International Conference on Artificial Intelligence and Statistics, volume 2, pages 6774, 2007.

[24] T. Kollig, A. Keller, *Efficient multidimensional sampling*, Computer Graphics Forum, vol. 21, no. 3, 2002.

[25] Martin D. Buhmann. *Radial Basis Functions: Theory and Implementations*. Cambridge University Press, July 2003. Google-Books-ID: TRMf53opzlsC.

[26] Donald Shepard. *A Two-dimensional Interpolation Function for Irregularly-spaced Data*, In Proceedings of the 1968 23rd ACM National Conference, ACM 68, pages 517524, New York, NY, USA, 1968.

[27] L. van der Maaten, *Accelerating t-sne using tree-based algorithms*, J. Mach. Learn. Res., vol. 15, pp. 32213245, 2014.

[28] Ching Tarn, Yinan Zhang and Ye Feng, *Sampling Clustering*, arXiv:1806.08245v1 [cs.CV], June 2018.

[29] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. *A Fast Learning Algorithm for Deep Belief Nets*, Neural Computation, 18(7):15271554, May 2006.

[30] B. Jacob, J. Chen, Y. Huang, I. Cohen, *Pearson correlation coefficient in Noise Reduction in Speech Processing*, Berlin, Germany:Springer-Verlag, pp. 1-4, 2009.

[31] D.C. Montgomery,*Design and Analysis of Experiments (3rd Edition)*, Wiley, New York (1991)

[32] L. J. Williams and H. Abdi, *Fishers least significance difference (LSD) test*, in Encyclopedia of Research Design. Thousand Oaks, 2010, pp. 491494.

[33] Y. Zhang, Y. Wang, S. Parthasarathy, "Visualizing attributed graphs via terrain metaphor", Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, pp. 1325-1334, 2017.

[34] Albert Kim , Eric Blais , Aditya Parameswaran , Piotr Indyk , Sam Madden , Ronitt Rubinfeld, Rapid sampling for visualizations with ordering guarantees, Proceedings of the VLDB Endowment, v.8 n.5, p.521-532, January 2015.