Degree project

Optimizing t-SNE using random sampling techniques

Author: Matej Buljan  
Supervisors: Jonas Nordqvist, Rafael M. Martins  
Examiner: Karl-Olof Lindahl  
Date: 2019-07-13  
Course Code: 2MA41E  
Subject: Mathematics  
Level: Bachelor

Department Of Mathematics
OPTIMIZING T-SNE USING RANDOM SAMPLING TECHNIQUES

MATEJ BULJAN

Abstract. The main topic of this thesis concerns t-SNE, a dimensionality reduction technique that has gained much popularity for showing great capability of preserving well-separated clusters from a high-dimensional space. Our goal with this thesis is twofold. Firstly we give an introduction to the use of dimensionality reduction techniques in visualization and, following recent research, show that t-SNE in particular is successful at preserving well-separated clusters. Secondly, we perform a thorough series of experiments that give us the ability to draw conclusions about the quality of embeddings from running t-SNE on samples of data using different sampling techniques. We are comparing pure random sampling, random walk sampling and so-called hubness sampling on a dataset, attempting to find a sampling method that is consistently better at preserving local information than simple random sampling. Throughout our testing, a specific variant of random walk sampling distinguished itself as a better alternative to pure random sampling.
CONTENTS

1. Introduction 3
1.1. Related work 4
2. Theoretical background 6
2.1. Dimensionality reduction 6
2.2. The t-SNE algorithm 7
2.3. Sampling 19
3. Methods 22
3.1. Experimental setup 22
3.2. Datasets 25
4. Results and analysis 26
5. Discussion and future work 35
References 37
1. Introduction

Dimensionality reduction techniques are widely used nowadays as tools for better understanding and interpretation of multivariate datasets. Their applications in the fields of machine learning and data science help researchers understand how objects with many observable attributes, which we may think of as points in multivariate datasets, interact with each other. This can be useful when attempting to discern groups or classes of objects that naturally appear in the high-dimensional space, by means of visualizing the data. The information gained can later be used for classifying future observations and gaining deeper insights into the inner structure of the dataset. Except for visualization, dimensionality reduction is used most often for feature selection and extraction, letting us to focus on the aspects of the dataset that are most important to us (see [JL+98], [BM01]).

Currently, a state-of-the-art dimensionality reduction technique is t-SNE, short for \textit{t-Distributed Stochastic Neighborhood Embedding}. The main objective of t-SNE is to preserve local information in high-dimensional space, i.e. preserving neighborhoods, when projected onto a low dimensional (usually two-dimensional) space. Its main drawback is that it is quite computationally heavy as it considers each point in a dataset separately in a quadratic ($O(N^2)$) way and thus takes a long time to perform its optimization.

It is often heard that we are living in the world of Big Data [Loh12]. By that term, we describe datasets, and often research and analysis of — or related to — those datasets, whose size, given in terms of the number of observations and features measured per observation, is \textit{large}. Therefore, analyzing or manipulating them requires not only professional-level hardware and software but also knowledge of many fields, specially mathematics and computer science. Big datasets are favorable when training machine learning models since having more (quality) input data means that we will be able to train a better, more realistic, more \textit{experienced} model. On the other hand, training models on big datasets can be extremely time-consuming, even with powerful hardware. This limits the ability to try out different types of models or different settings in order to optimize the existing model.

A number of optimization techniques involving sampling have been proposed in order to try and fix that issue. However, by using less data, we are inherently losing some information and so the overall quality of the model will suffer. The assumption when employing a sampling technique when building a machine learning model is that if we are able to take a sample that is highly representative of the entire
dataset and train our model on that sample, the end result would give *reasonably*
good results (compared to training on the entire dataset) while taking *significantly*
less time to train the model. For visualization purposes, this translates to finding
the points which are most representative of the feature set that we want to preserve
when applying dimensionality reduction techniques.

It is worth to remember that the idea of sampling has been one of the most com-
mon concepts in statistics ever since its inception. Our incapability to gather data
on entire populations made sampling and inferential statistics not only important
but necessary. The same idea motivated us to explore the effects of sampling in a
study of dimensionality reduction of big datasets.

The aim of this thesis is twofold: firstly we explain the t-SNE algorithm and show
that it is able to preserve well-separated clusters and secondly, study the differences
when applying four different sampling techniques to the t-SNE algorithm. The main
research question that we will try to answer is if there is a sampling technique that is
better suited for dimensionality reduction than pure random sampling. Therefore,
we will be applying traditional random sampling, two versions of random walk
sampling and sampling by selection of most representative points (“hubs”) to a
dataset, then run the t-SNE algorithm on the samples and compare the results by
calculating three different quality measures for each embedding. The experiments
are run multiple times, for different sample sizes and across multiple datasets to
draw conclusions about the effectiveness of the sampling techniques being tested.

In §2, we present the theory needed for understanding how the t-SNE algorithm
works as well as show that it is indeed successful at producing a full visualization
of data while preserving clusters from the high-dimensional space. §3 describes
the methods used for our comparison of the different sampling techniques while
in §4 the results of our testing are given and analyzed. In §5, a discussion is
made about possible points of improvement to the experimental setup, conclusions
reached through the testing and finally ways to expand the research part of this
thesis into future work.

1.1. Related work. A DR technique called *Stochastic neighborhood embedding*
(SNE) [HR03] served as a base, which was modified to create t-SNE, presented
in 2008 [vdMH08]. Since then, it has been used as the go-to visualization tool.
However, it’s best features, such as cluster preservation and the “promise” of a
successful visualization haven’t been proved mathematically until 2017 and 2018.
The paper [LS17] proves the preservation and contraction of clusters when doing
DR while *successful visualization* is proved in [AHK18] (all of those results will be further analyzed in the thesis).

The idea of sampling from manifolds is explored in [ÖAG10]. For a more mathematical approach to sampling from graphs, we turned to [Lov96] while writing this thesis. Finally, our prototype algorithm for random walk sampling comes from [BJ17] while hubness sampling is done here employing the same idea as in [TRMI14].
2. Theoretical background

In this section we provide a theoretical background to the topic of the thesis. In §2.1 we discuss dimensionality reduction, in §2.2 we discuss the t-SNE algorithm and in §2.3 we discuss the different sampling techniques being compared in this thesis.

2.1. Dimensionality reduction. Machine learning is the scientific study of algorithms and statistical models that computer systems use to effectively perform a specific task without using explicit instructions, relying on patterns and inference instead (see for example [LRU14]). It is seen as a subset of artificial intelligence. Based on the type of learning, it can fundamentally be divided into supervised and unsupervised learning. In contrast to supervised learning in which a model is trained using labeled data, unsupervised learning is about extracting (useful) information from unlabeled datasets. Dimensionality reduction techniques (DR techniques for short) are an integral part of the domain of unsupervised learning. One of DR techniques’ primary uses is in visualization of multivariate datasets. A typical usage scenario might involve trying to make a two- or three-dimensional visualization of a dataset that is originally fully described in \( \mathbb{R}^d \) where \( d \gg 3 \). Of course it is impossible to preserve all information that was conveyed in the high-dimensional space when applying DR (meaning primarily the pairwise distances between points which can not all be preserved simultaneously), however the end result might still be useful enough to yield information about the inner structure of the dataset. It is important to know exactly which aspect of the data we want to capture best because that is what our choice of DR technique depends most on. A comparison of some of the most used techniques, along with their descriptions is given in [vdMPvdH09].

Perhaps the most heavily studied and well-known is principal component analysis (PCA). PCA is a linear mapping technique that does an orthogonal projection of the data onto a space spanned by the \( q \) eigenvectors of the datasets’ covariance matrix that are associated to the \( q \) largest eigenvalues, where \( q \) is usually 2 or 3. The main goal is to capture as much variance as the data can provide. The method was proposed by Pearson in 1901 [F.R01], but the idea has been around for even longer. It has been heavily studied and its properties and uses are well-known

---

\(^1\)In machine learning, the notion of training a model refers to modifying a certain assumed model so as to improve its performance with time. The performance improvement is often contained in minimizing a preset loss function.
by now. The main advantages are that it is simple and fast to implement and run. However, depending on the purpose of the DR, PCA may be insufficient in reducing high-dimensional data as it may not yield much information in terms of visualization. Also, it is focused on preserving variance which is in practice seldom considered the most useful feature of the dataset, see [vdMPvdH09].

2.2. The t-SNE algorithm. We begin by giving the definition of the Kullback-Leibler divergence, specifically the version for discrete probability distributions. This is a key concept that we will need for better understanding of the t-SNE algorithm.

Definition 1 (Kullback-Leibler divergence as defined in [MMK03]). Let $P$ and $Q$ be discrete probability distributions defined on the same probability space $\mathcal{X}$ such that $Q(x) = 0 \implies P(x) = 0 \ \forall x \in \mathcal{X}$. Then the Kullback-Leibler divergence from

\begin{align*}
\text{Kullback-Leibler divergence} = \sum_{x \in \mathcal{X}} P(x) \log \frac{P(x)}{Q(x)}.
\end{align*}

Figure 1. A dimensionality reduction comparison of PCA and t-SNE applied to a mixture of seven six-dimensional Gaussian datasets.

As contrast to PCA, we have t-distributed stochastic neighborhood embedding, typically abbreviated t-SNE. The method was presented in [vdMH08] and it is a non-linear mapping focused on retaining the local structure of data in the map. This means that clusters\(^2\) from a high-dimensional space should be clearly visible in the low-dimensional space. For a comparison of dimensionality reduction between PCA and t-SNE, see Figure 1.
$Q$ to $P$ is defined as

$$KL(P\|Q) = \sum_{x \in X} P(x) \log \left( \frac{P(x)}{Q(x)} \right).$$

**Remark.** For the interested reader, the definition of the Kullback-Leibler divergence for continuous probability distributions is given in [Bis06, p.55].

It is worth noting that the Kullback-Leibler divergence is not a metric: it does not obey the triangle inequality and, in general $KL(P\|Q) \neq KL(Q\|P)$. Expressed in the language of Bayesian inference, $KL(P\|Q)$ is a measure of the information gained when one revises one’s beliefs from the prior probability distribution $Q$ to the posterior probability distribution $P$. In other words, it is the amount of information lost when $Q$ is used to approximate $P$.

The logarithm in the formula is taken to base 2, hence measuring the divergence in bits.

**Definition 2.** Perplexity of a discrete probability distribution $f$ is defined as $2^{H(f)}$ where $H(f) = -\sum_x f(x) \log_2 f(x)$ is known as the entropy of the distribution. It is not necessary that the base of exponentiation is 2 as long as it is the same as the base of the logarithm in the exponent.

As stated in the introduction, the t-SNE algorithm is based on SNE. In this section we will introduce the main features of SNE and explain how the modifications that were done to it (until t-SNE was finally created) impacted its quality of output and time-efficiency.

We denote by $X = \{x_1, x_2, \ldots, x_N\} \subset \mathbb{R}^d$ the $d$-dimensional input dataset. Let $s$ be an integer such that $s \ll d$, the t-SNE algorithm computes an $s$-dimensional embedding $Y = \{y_1, y_2, \ldots, y_N\} \subset \mathbb{R}^s$ of the points in $X$. The most common choice for DR is $s = 2$ or $s = 3$.

Throughout this thesis we denote by

$$\| \cdot \| := \| \cdot \|_2$$

the $L^2$ (Euclidean) norm.

SNE starts by converting the high-dimensional Euclidean distances between datapoints into conditional probabilities that represent similarities. The similarity of datapoint $x_j$ to datapoint $x_i$ is the conditional probability, $p_{ji}$, that $x_i$ would pick $x_j$ as its neighbor if neighbors were picked in proportion to their probability density under a Gaussian centered at $x_i$. For nearby datapoints, $p_{ji}$ is relatively high,
whereas for widely separated datapoints, \( p_{ij|i} \) will be almost infinitesimal (for reasonable values of the variance of the Gaussian, \( \sigma_i \)). The conditional probability \( p_{ij} \) is computed as:

\[
p_{ij} = \frac{\exp(-\|x_i - x_j\|^2/2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|x_i - x_k\|^2/2\sigma_i^2)}
\]

where the \( \sigma_i \) are chosen in such a way that the perplexity of the conditional distribution across all data points given \( x_i \), \( f_i \) matches a user-defined value. For the low-dimensional counterparts \( y_i \) and \( y_j \) of the high-dimensional datapoints \( x_i \) and \( x_j \), it is possible to compute a similar conditional probability, which we denote by \( q_{ij|i} \):

\[
q_{ij} = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{k \neq l} \exp(-\|y_k - y_l\|^2)}.
\]

It is needed to point out that \( p_{ii} = 0 \) and \( q_{ii} = 0 \) \( \forall i \). If the points \( y_i, y_j \in Y \) correctly model the similarity between the high-dimensional datapoints \( x_i, x_j \in X \), the conditional probabilities \( p_{jj|i} \) and \( q_{jj|i} \) will be equal. Motivated by this observation, SNE aims to find an embedding that minimizes the mismatch between \( p_{jj|i} \) and \( q_{jj|i} \). Recalling that the Kullback-Leibler divergence from \( Q \) to \( P \) is a natural measure of the faithfulness with which \( q_{ij|i} \) models \( p_{ij|i} \), SNE minimizes the sum of Kullback-Leibler divergences over all datapoints using gradient descent. The cost function \( C \) is given by

\[
C = \sum_i KL(P_i\|Q_i) = \sum_i \sum_j p_{ij|i} \log \frac{p_{ij|i}}{q_{ij|i}}
\]

where \( P_i \) represents the conditional probability distribution over all other datapoints given datapoint \( x_i \), and \( Q_i \) represents the conditional probability distribution over all other map points given point \( y_i \in Y \).

Symmetric SNE has the property that \( p_{ij} = p_{ji} \) and \( q_{ij} = q_{ji} \), which allows for a simpler gradient descent to the cost function \( C \), effectively making the calculations faster (and even give slightly better results, as shown empirically). The high- and low-dimensional affinities are now defined as

\[
p_{ij} = \frac{p_{ij|i} + p_{ji|i}}{2N} \quad q_{ij} = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{k \neq l} \exp(-\|y_k - y_l\|^2)}
\]

respectively, while the cost function \( C \) is calculated as a single Kullback-Leibler divergence between two joint probability distributions \( P \) and \( Q \):

\[
C = KL(P\|Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}.
\]
A problem known as the “crowding problem”, which is characteristic of many multidimensional scaling techniques, including SNE, is being alleviated in t-SNE by using a heavy-tailed Student t-distribution with one degree of freedom (which is the same as the Cauchy distribution) for low-dimensional affinities $q_{ij}$:

\[
q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq l} (1 + \|y_i - y_k\|^2)^{-1}}.
\] (2.4)

This also speeds up the calculations as this type of probability is easier to calculate than the one including exponentials. For more information about the crowding problem and the motivation as to why this specific choice of distribution solves that problem, we point the reader to the original t-SNE paper [vdMH08].

Given a high-dimensional dataset $X$, t-SNE first computes the pairwise affinities $p_{ij}$ in the same way as Symmetric SNE. The points in the low-dimensional space $Y$ are initialized randomly from a Gaussian distribution $\mathcal{N}(0, 10^{-4}I)$, where $I$ is the $s$-dimensional identity matrix. Then the initial low-dimensional affinities $q_{ij}$ are calculated as given in (2.4). The objective of t-SNE is to minimize the cost function $C(Y)$ that was given in (2.3) for Symmetric SNE, using gradient descent.

**Lemma 2.1.** Gradient to the cost function $C(Y)$ defined as the Kullback-Leibler divergence

\[
C(Y) = KL(P\|Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}
\]

is given by

\[
\frac{\partial C}{\partial y_i} = 4 \sum_j (p_{ij} - q_{ij})(1 + \|y_i - y_j\|^2)^{-1}(y_i - y_j).
\] (2.5)

By studying the gradient descent as a discrete dynamical system, we can derive a proof for preserving clusters. This is done in the next section.

**Proof of Lemma 2.1.** Put

\[
d_{jk} := \|y_j - y_k\|, \quad f_{jk} := (1 + d_{jk}^2)^{-1}, \quad Z := \sum_{\ell \neq m} f_{\ell m}.
\] (2.6)

To simplify notation, we introduce the following notation

\[
\sum_j := \sum_{j=1}^N, \quad \sum_{j,k} := \sum_{j=1}^N \sum_{k=1}^N,
\]

First we note that $\frac{\partial f_{\ell m}}{\partial d_{kl}} = 0$ unless $i = k, j = l$. Hence, by the chain rule, we obtain

\[
\frac{\partial C}{\partial y_i} = \sum_{j,k} \frac{\partial C}{\partial q_{jk}} \sum_{\ell,m} \frac{\partial q_{jk}}{\partial f_{\ell m}} \frac{\partial f_{\ell m}}{\partial d_{\ell m}} \frac{\partial d_{\ell m}}{\partial y_i}.
\]
We recall the definition of the Kullback-Leibler divergence

\[ C = \sum_{j,k} p_{jk} \log \left( \frac{p_{jk}}{q_{jk}} \right) = \sum_{j,k} p_{jk} \left( \log(p_{jk}) - \log(q_{jk}) \right). \]

Thus, we have the partial derivative of \( C \) with respect to \( q_{jk} \) given by

\[ \frac{\partial C}{\partial q_{jk}} = -\frac{p_{jk}}{q_{jk}}. \]

Hence, the previous equation is now given by

\[ \partial C \partial y_i = -\sum_{j,k} p_{jk} \sum_{\ell,m} \frac{\partial q_{jk}}{\partial f_{\ell m}} \frac{\partial d_{\ell m}}{\partial y_i}. \]

Further, note that \( \frac{\partial d_{\ell m}}{\partial y_i} = 0 \) unless \( \ell = i \) or \( k = i \). Thus, we obtain

\[ (2.7) \quad \frac{\partial C}{\partial y_i} = -\left( \sum_{j,k} \frac{p_{jk}}{q_{jk}} \sum_{\ell} \frac{\partial q_{jk}}{\partial f_{i \ell}} \frac{\partial f_{i \ell}}{\partial d_{i \ell}} \frac{\partial d_{i \ell}}{\partial y_i} + \sum_{j,k} \frac{p_{jk}}{q_{jk}} \sum_{m} \frac{\partial q_{jk}}{\partial f_{i m}} \frac{\partial f_{i m}}{\partial d_{i m}} \frac{\partial d_{i m}}{\partial y_i} \right). \]

Moreover, since the arguments of \( d \) and \( f \) commute, we obtain

\[ \frac{\partial C}{\partial y_i} = -2 \sum_{j,k} \frac{p_{jk}}{q_{jk}} \sum_{\ell} \frac{\partial q_{jk}}{\partial f_{i \ell}} \frac{\partial f_{i \ell}}{\partial d_{i \ell}} \frac{\partial d_{i \ell}}{\partial y_i}. \]

Rearranging yields

\[ (2.8) \quad \frac{\partial C}{\partial y_i} = -2 \sum_{\ell} \left( \sum_{j,k} \frac{p_{jk}}{q_{jk}} \frac{\partial q_{jk}}{\partial f_{i \ell}} \right) \frac{\partial f_{i \ell}}{\partial d_{i \ell}} \frac{\partial d_{i \ell}}{\partial y_i}. \]

Computing the partial derivatives yields

\[ (2.9) \quad \frac{\partial f_{i \ell}}{\partial d_{i \ell}} = -\frac{2d_{i \ell}}{(1 + d_{i \ell})^2} = -2d_{i \ell}f_{i \ell}^2 = -2d_{i \ell}Z^2q_{i \ell}^2 \]

and

\[ (2.10) \quad \frac{\partial d_{i \ell}}{\partial y_i} = \frac{1}{d_{i \ell}}(y_i - y_{\ell}). \]

Insertion of (2.9) and (2.10) in (2.8) yields

\[ (2.11) \quad \frac{\partial C}{\partial y_i} = -4 \sum_{\ell} \left( \sum_{j,k} \frac{p_{jk}}{q_{jk}} \frac{\partial q_{jk}}{\partial f_{i \ell}} \right) Z^2q_{i \ell}^2(y_i - y_{\ell}). \]

Moreover, due to the definition of \( q_{jk} \) including both the factor \( f_{jk} \), and the sum of all terms \( Z = \sum_{\ell \neq m} f_{\ell m} \) in the denominator, we obtain the partial derivatives

\[ (2.12) \quad \frac{\partial q_{jk}}{\partial f_{jk}} = \frac{Z - f_{jk}}{Z^2} = \frac{1}{Z}(1 - q_{jk}) \quad \text{and} \quad \frac{\partial f_{\ell m}}{\partial f_{jk}} = -\frac{f_{\ell m}}{Z^2} = -\frac{q_{\ell m}}{Z}. \]

Insertion of (2.12) yields

\[ (2.13) \quad \frac{\partial C}{\partial y_i} = -4 \sum_{\ell} \frac{1}{Z} \left( \sum_{j,k} \frac{p_{jk}}{q_{jk}} - \sum_{j,k} \frac{p_{jk}}{q_{jk}} \right) Z^2q_{i \ell}^2(y_i - y_{\ell}). \]
This, together with $\sum_{j,k} p_{jk} = 1$ in (2.13) yields
\[
\frac{\partial C}{\partial y_i} = -4 \sum_{\ell} \frac{1}{Z} \left( -\frac{p_{\ell \ell}}{q_{\ell \ell}} + \sum_{j,k} p_{jk} \right) Z^2 q_{\ell \ell}^2 (y_i - y_{\ell \ell})
\]
\[
= -4 \sum_{\ell} \frac{1}{Z} \left( -\frac{p_{\ell \ell}}{q_{\ell \ell}} + 1 \right) Z^2 q_{\ell \ell}^2 (y_i - y_{\ell \ell})
\]
\[
= -4 \sum_{\ell} \left( -p_{\ell \ell} + q_{\ell \ell} \right) Z q_{\ell \ell} (y_i - y_{\ell \ell})
\]
\[
= 4 \sum_{\ell} \left( p_{\ell \ell} - q_{\ell \ell} \right) (1 + \|y_i - y_{\ell \ell}\|^2)^{-1} (y_i - y_{\ell \ell}).
\]

This completes the proof of the lemma. \( \square \)

The proof presented above is a somewhat expanded version of the proof of that is to be found in [vdMH08, Appendix A]. Specifically, we expanded the calculations presented in the original paper by bringing attention to how the specific derivatives are calculated through the chain rule. In addition, we included a number of middle-steps in the algebraic manipulations that were omitted in the original proof.

Multiplying the pairwise affinities for $\mathbb{R}^d$ by a user-defined exaggeration constant $\alpha > 1$ in the first $m$ learning iterations of gradient descent accelerates convergence in the early stages of optimization. This technique is called early exaggeration. See Section 3 in the original paper on t-SNE [vdMH08, §3] for more details regarding the algorithm.

As mentioned earlier, the main focus of t-SNE is preserving local neighborhood information. The method produces an $s$-dimensional embedding such that the points in the same clusters are noticeably closer together compared with points in different clusters. In a recent paper published by Arora, Hu and Kothari [AHK18], the authors define rigorously the concept of visualization, to prove that t-SNE does in fact manage to successfully produce low-dimensional embeddings of well-studied probabilistic generative models for clusterable data. As an example, Gaussian mixtures in $\mathbb{R}^d$ are with high probability visualized in two dimensions. They also compare t-SNE in that regard to some classic DR techniques like the aforementioned PCA which shows its weakness by not being able to produce an embedding with clearly visible and well-separated clusters (see FIGURE 1). A first step into proving that t-SNE is able to recover well-separated clusters is given by Linderman and Steinerberger in [LS17]. Their analysis is focused on the early exaggeration
phase of t-SNE. Here we will present proofs of two lemmas from the area of discrete dynamical systems that are used in the referenced paper to prove the main result.

2.2.1. Cluster preserving. The results are formally stated here for a set of points \( \{x_1, \ldots, x_N\} \) and a set of mutual affinities \( p_{ij} \) which need not be obtained using the standard t-SNE normalizations, but only using a set of three assumptions. In [LS17, §3.2] the first assumption encapsulates the notion of a clustered dataset. Suppose there exists a positive integer \( k \in \mathbb{N}^+ \) and a map \( \kappa: \{x_1, \ldots, x_N\} \rightarrow \{A_1, A_2, \ldots, A_k\} \) assigning each point to one of the \( k \) clusters \( A_\ell \), where \( \ell = 1, 2, \ldots, k \) such that the following property holds: if \( x_i, x_j \in A_\ell \), then

\[
p_{ij} \geq \frac{1}{10N|A_\ell|},
\]

where \( |A_\ell| \) is the size of the cluster in which \( x_i \) and \( x_j \) lie.

We denote the step size used in the gradient descent as \( h > 0 \). The second assumption needed to prove their results is regarding the parameter choice. Namely, we assume that \( \alpha \) and \( h \) are chosen such that, for some \( 1 \leq i \leq n \)

\[
\frac{1}{100} \leq \alpha h \sum_{j \neq i \text{ same cluster}} p_{ij} \leq \frac{9}{10}.
\]

The last assumption is that the initial embedding satisfies \( Y(0) \in [-0.01, 0.01]^2 \).

Now we introduce a type of discrete dynamical system on sets of points in \( \mathbb{R}^s \) and describe their asymptotic behaviour. Let \( B_\varepsilon(x) \) denote the ball of radius \( \varepsilon \) centered at \( x \). We denote also \( A + B = \{a + b : a \in A \land b \in B\} \).

**Definition 3.** Let \( m \geq 2 \) be a positive integer, and let \( S = \{z_1, \ldots, z_m\} \subseteq \mathbb{R}^n \). We define the **convex hull** of \( S \) as the convex combination of all points in \( S \), i.e. all points of the form

\[
\alpha_1 z_1 + \cdots + \alpha_m z_m,
\]

where \( \alpha_i \geq 0 \), and \( \sum_i \alpha_i = 1 \). We denote the convex hull of \( S \) by \( \text{conv} S \).

**Definition 4.** The **diameter** of a subset of \( \mathbb{R}^n \) is defined as

\[
\text{diam}\{z_1, \ldots, z_m\} = \max_{i,j} \|z_i - z_j\|.
\]

**Lemma 2.2** (Stability of the convex hull, [Lemma 1, [LS17]]). Define for each integer \( i \in \{1, \ldots, n\} \), \( z_i(0) := z_i \), and for each integer \( t \geq 1 \) define \( z_i(t) \) recursively by

\[
z_i(t + 1) := z_i(t) + \sum_{j=1}^n \alpha_{i,j,t}(z_j(t) - z_i(t)) + \varepsilon_i(t).
\]
Moreover, suppose there is a uniform upper bound on the coefficients $\alpha$ and the error term $\varepsilon$

$$\sum_{j=1}^{n} \alpha_{i,j,t} \leq 1 \quad \text{and} \quad \|\varepsilon_{i}(t)\| \leq \varepsilon,$$

and a uniform lower bound on the coefficients for all $t \geq 1$ and $i \neq j$

$$\alpha_{i,j,t} \geq \delta > 0.$$

Then

$$\text{conv}\{z_{1}(t+1), z_{2}(t+1), \ldots, z_{n}(t+1)\} \subseteq \text{conv}\{z_{1}(t), z_{2}(t), \ldots, z_{n}(t)\} \pm B_{\varepsilon}(0).$$

**Proof of Lemma 2.2.** We note that

$$z_{i}(t+1) = z_{i}(t) + \sum_{j=1}^{n} \alpha_{i,j,t}(z_{j}(t) - z_{i}(t)) + \varepsilon_{i}(t)$$

(2.14)

$$= \left(1 - \sum_{j=1}^{n} \alpha_{i,j,t} \right) z_{i}(t) + \sum_{j=1}^{n} \alpha_{i,j,t} z_{j}(t) + \varepsilon_{i}(t).$$

For all $j \in \{1, \ldots, n\}$, denote by $\beta_{j}$ the coefficient of $z_{j}(t)$ in (2.14), then the following is satisfied

$$\sum_{j=1}^{n} \beta_{j} = \left(1 - \sum_{j=1}^{n} \alpha_{i,j,t} \right) + \sum_{j=1}^{n} \alpha_{i,j,t} = 1.$$

Hence, by the definition of the convex hull we have obtained $z_{i}(t+1) - \varepsilon_{i}(t) \in \text{conv}\{z_{1}(t), z_{2}(t), \ldots, z_{n}(t)\}$, and applying this for every $i \in \{1, \ldots, n\}$ gives the sought results. \qed

The above proof was written as a more clarified version of the proof presented in the original paper. Namely, we brought more attention to how exactly the sum of coefficients in (2.14) shows that $z_{i}(t+1)$ is contained within the convex hull defined by the $z_{i}(t), i = 1, \ldots, n$.

**Lemma 2.3** (Contraction inequality, [LS17, Lemma 2]). With the notation from Lemma 2.2, if the diameter is large

$$\text{diam}\{z_{1}(t), z_{2}(t), \ldots, z_{n}(t)\} \geq \frac{10\varepsilon}{n\delta},$$

then

$$\text{diam}\{z_{1}(t+1), z_{2}(t+1), \ldots, z_{n}(t+1)\} \leq \left(1 - \frac{n\delta}{20}\right) \text{diam}\{z_{1}(t), z_{2}(t), \ldots, z_{n}(t)\}.$$
Proof of Lemma 2.3. The diameter of a convex hull is preserved when projecting the set of points onto the line that connects the two points whose distance is equal to the diameter. We will now show that the lemma holds even when projecting the set of points onto an arbitrary line, which will imply the desired result. We may without loss of generality use the projection $\pi_x : \mathbb{R}^d \to \mathbb{R}$ that projects the points onto the $x$-axis i.e. taking only the first coordinate of each point. Let us abbreviate the diameter of the projection:

$$d(t) := \text{diam}\{\pi_x z_1(t), \pi_x z_2(t), \ldots, \pi_x z_n(t)\}$$

which is constant during time $t$. Translating the set to the origin, we may w.l.o.g. assume that this set is contained in $\{\pi_x z_1(t), \pi_x z_2(t), \ldots, \pi_x z_n(t)\} \subset [0, d(t)]$. We can then subdivide the interval into two regions $I_1 = \left[0, \frac{d(t)}{2}\right]$ and $I_2 = \left(\frac{d(t)}{2}, d(t)\right]$ and denote the number of points in each interval by $i_1, i_2$. Since $i_1 + i_2 = n$, it is clear that either $i_1 \geq n/2$ or $i_2 \geq n/2$. We assume without loss of generality that the first case holds. Projections are linear, thus

$$\pi_x z_i(t + 1) = \pi_x z_i(t) + \sum_{j=1}^{n} \alpha_{i,j,t} \pi_x (z_j(t) - z_i(t)) + \pi_x \epsilon_i(t).$$

Let us use $\sigma$ as the sum of all coefficients

$$0 \leq \sigma := \sum_{j=1}^{n} \alpha_{i,j,t} \leq 1.$$  

We may divide the sum from (2.15) according to the regions $I_1, I_2$

$$S := \sum_{j=1}^{n} \alpha_{i,j,t} \pi_x (z_j(t) - z_i(t)) = \sum_{\pi_x z_j \leq d(t)/2} \alpha_{i,j,t} \pi_x (z_j(t) - z_i(t))$$

$$+ \sum_{\pi_x z_j > d(t)/2} \alpha_{i,j,t} \pi_x (z_j(t) - z_i(t)).$$

Again using the fact that $\pi_x$ is linear and by taking the largest possible values for $\pi_x(z_j(t))$, we have

$$S \leq \sum_{\pi_x z_j \leq d(t)/2} \alpha_{i,j,t} \left(\frac{d(t)}{2} - \pi_x z_i(t)\right) + \sum_{\pi_x z_j > d(t)/2} \alpha_{i,j,t} (d(t) - \pi_x z_i(t)) =: S'.$$

Using (2.16), we obtain

$$S' = \sum_{\pi_x z_j \leq d(t)/2} \alpha_{i,j,t} \frac{d(t)}{2} + \sum_{\pi_x z_j > d(t)/2} \alpha_{i,j,t} d(t) - \sigma \pi_x z_i(t).$$
Furthermore, we have

\[
\left( \frac{1}{2} \sum_{\pi_x z_j \leq d(t)/2} \alpha_{i,j,t} + \sum_{\pi_x z_j > d(t)/2} \alpha_{i,j,t} \right) d(t) = \left( \frac{1}{2} \sum_{\pi_x z_j \leq d(t)/2} \alpha_{i,j,t} + \left( \sigma - \sum_{\pi_x z_j \leq d(t)/2} \alpha_{i,j,t} \right) \right) d(t) = \left( \sigma - \frac{1}{2} \sum_{\pi_x z_j \leq d(t)/2} \alpha_{i,j,t} \right) d(t).
\]

Moreover, using the lower bound \( \alpha_{i,j,t} \geq \delta \) and remembering that for \( I_2 \), \( i_2 < \frac{n}{2} \) we have

\[
(2.20) \quad \left( \sigma - \frac{1}{2} \sum_{\pi_x z_j \leq d(t)/2} \alpha_{i,j,t} \right) d(t) \leq \left( \sigma - \frac{n\delta}{4} \right) d(t).
\]

Combining the results of (2.17), (2.18) and (2.19) with (2.20), as well as ignoring for now the error term in (2.15), we have

\[
\pi_x z_i(t + 1) = \pi_x z_i(t) + \sum_{j=1}^{n} \alpha_{i,j,t} \pi_x (z_j(t) - z_i(t)) 
\leq (1 - \sigma) \pi_x z_i(t) + \left( \sigma - \frac{n\delta}{4} \right) d(t)
\]

Taking now the maximum value for \( \pi_x z_i(t) \),

\[
(1 - \sigma) \pi_x z_i(t) + \left( \sigma - \frac{n\delta}{4} \right) d(t) \leq (1 - \sigma) d(t) + \left( \sigma - \frac{n\delta}{4} \right) d(t) = \left( 1 - \frac{n\delta}{4} \right) d(t),
\]

which shows that \( \pi_x z_i(t + 1) \in [0, d(t)(1 - n\delta/4)] \). Accounting for the error term, we get

\[
d(t + 1) \leq \left( 1 - \frac{n\delta}{4} \right) d(t) + 2\varepsilon.
\]

If the diameter is indeed disproportionately large

\[
d(t) \geq \frac{10\varepsilon}{n\delta},
\]

then this can be rearranged as

\[
\varepsilon \leq \frac{n\delta}{10} d(t)
\]

and therefore

\[
\left( 1 - \frac{n\delta}{4} \right) d(t) + 2\varepsilon \leq \left( 1 - \frac{n\delta}{4} \right) d(t) + \frac{n\delta}{5} d(t) \leq \left( 1 - \frac{n\delta}{20} \right) d(t).
\]

Since this is true in every projection, it also holds for the diameter of the original set. This completes the proof of the Lemma.
In the proof above we decided to use different wording compared to the proof from the original paper, along with adding several shorter comments in the interest of clarifying the reasoning used in the proof, but not deviating from the original sketch of the proof.

The main result of the Linderman and Steinerberger paper [LS17] is that the gradient descent of t-SNE acting on one particular cluster of a dataset can be rewritten as a dynamical system of the type discussed previously which therefore proves the preserving of clusters.

**Theorem 2.4** (Cluster preserving theorem from [LS17]). The diameter of the embedded cluster \( A_\ell \) decays exponentially (at universal rate) until its diameter satisfies, for some universal constant \( c > 0 \),

\[
\text{diam}\{A_\ell\} \leq c \cdot h \left( \alpha \sum_{j \neq i} p_{ij} + \frac{1}{n} \right).
\]

**Proof.** We start by showing that the \( q_{ij} \) are comparable as long as the point set is contained in a small region. Let now \( \{y_1, y_2, \ldots, y_n\} \subset [-0.02, 0.02]^2 \) and recall the definitions

\[
q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq l}(1 + \|y_l - y_k\|^2)^{-1}}, \quad \text{and} \quad Z = \sum_{k \neq l}(1 + \|y_k - y_i\|^2)^{-1}.
\]

Then, however, it is easy to see that \( 0 \leq \|y_i - y_j\| \leq 0.06 \) implies

\[
\frac{9}{10} \leq q_{ij} Z = (1 + \|y_i - y_j\|^2)^{-1} \leq 1.
\]

We will now restrict ourselves to a small embedded cluster \( A_m \) and rewrite the gradient descent method as

\[
y_i(t + 1) = y_i(t) + \sum_{j \neq i} (\alpha h)p_{ij} q_{ij} Z(y_j(t) - y_i(t)) + \sum_{j \neq i} (\alpha h)p_{ij} q_{ij} Z(y_j(t) - y_i(t)) - h \sum_{j \neq i} q_{ij}^2 Z(y_j(t) - y_i(t)),
\]

where the first sum is yielding the main contribution and the other two sums are treated as a small error. Applying our results for dynamical systems of this type requires us to verify the conditions. We start by showing the conditions on the coefficients to be valid. Clearly,

\[
\alpha h p_{ij} q_{ij} Z \geq \frac{9}{10} \geq \frac{9}{100} n |A_m| \sim \delta.
\]
which is clearly admissible whenever $\alpha h \sim n$. As for the upper bound, it is easy to see that

$$\sum_{j \neq i}^{\text{same cluster}} (\alpha h)p_{ij}q_{ij}Z \leq \alpha h \sum_{j \neq i}^{\text{same cluster}} p_{ij} \leq 1.$$  

It remains to study the size of the error term for which we use the triangle inequality

$$\left\| \sum_{j \neq i}^{\text{other clusters}} (\alpha h)p_{ij}q_{ij}Z(y_j(t) - y_i(t)) \right\| \leq \sum_{j \neq i}^{\text{other clusters}} p_{ij} \|y_j(t) - y_i(t)\|$$

$$\leq 0.06\alpha h \sum_{j \neq i}^{\text{other clusters}} p_{ij}$$

and, similarly for the second term,

$$\left\| h \sum_{j \neq i} q_{ij}^2 Z(y_j(t) - y_i(t)) \right\| \leq h \sum_{j \neq i} q_{ij} \|y_j(t) - y_i(t)\| \leq 0.06h \sum_{j \neq i} q_{ij} \leq \frac{0.1h}{n}.$$  

This tells us that the norm of the error term is bounded by

$$\|\varepsilon\| \leq 0.1h \left( \alpha \sum_{j \neq i}^{\text{other clusters}} p_{ij} + \frac{1}{n} \right).$$

It remains to check whether time-scales fit. The number of iterations $\ell$ for which the assumption $Y^{\ell}(0) \subset [-0.02, 0.02]^2$ is reasonable is at least $\ell \geq 0.01/\varepsilon$. At the same time, the contraction inequality implies that in that time the cluster shrinks to the size

$$\max \left\{ \frac{10\varepsilon}{|A_m|^\delta}, 0.01 \left( 1 - \frac{1}{20} \right)^\ell \right\} \leq \max \left\{ \frac{10\varepsilon}{|A_m|^\delta}, 8\varepsilon \right\},$$

where the last inequality follows from the elementary inequality

$$\left( 1 - \frac{1}{20} \right)^{1/100\varepsilon} \leq 8\varepsilon.$$

\[ \square \]

The presented proof can also be found in [LS17, §6]. We encourage the reader to see the full paper for additional remarks concerning the generality of the proof and its applications.

A potential pitfall is that this result only guarantees the preserving of each cluster for itself, but not excluding the case where a number of “preserved” clusters are in fact overlapping which would not give a successful visualization. However, this has been taken care of by Arora, Hu and Kothari in [AHK18] by keeping track of the centroids of the clusters. To say more about their results, we must first define
concepts such as full visualization and well-separated, spherical data in the way presented in their paper. For shorter notation, the authors of the paper use \([n]\) to denote the set \([1, 2, \ldots, n]\).

**Definition 5** (Visualization, as defined in [AHK18]). Let \(Y\) be a 2-dimensional embedding of a dataset \(X\) with ground-truth-clustering \(C_1, C_2, \ldots, C_k\). Given \(\epsilon \geq 0\), we say that \(Y\) is a \((1-\epsilon)\)-visualization of \(X\) if there exists a partition \(P_1, P_2, \ldots, P_k, P_{err}\) of \([n]\) such that:

1. For each \(i \in [k]\), \(P_i\) \((1-\epsilon)\)-visualizes \(C_i\) in \(Y\) and
2. \(|P_{err}| \leq \epsilon n\).

In particular, when \(\epsilon = 0\), we say that \(Y\) is a full visualization of \(X\).

**Definition 6** (Well-separated, spherical data, as defined in [AHK18]). Let \(X = \{x_1, x_2, \ldots, x_n\} \subset \mathbb{R}^d\) be clusterable data with \(C_1, C_2, \ldots, C_k\) defining the individual clusters such that for each \(\ell \in [k]\), \(|C_\ell| \geq 0.1(n/k)\). We say that \(X\) is \(\gamma\)-spherical and \(\gamma\)-well-separated if for some \(b_1, b_2, \ldots, b_k > 0\), we have:

1. **\(\gamma\)-Spherical:** For any \(\ell \in [k]\) and \(i, j \in C_\ell (i \neq j)\), we have \(\|x_i - x_j\|^2 \geq \frac{b_\ell}{1 + \gamma}\), and for any \(i \in C_\ell\) we have \(|\{j \in C_\ell \setminus \{i\} : \|x_i - x_j\|^2 \leq b_\ell\}| \geq 0.51|C_\ell|\).
2. **\(\gamma\)-Well-separated:** For any \(\ell, \ell' \in [k] (\ell \neq \ell')\), \(i \in C_\ell\) and \(j \in C_{\ell'}\) we have \(\|x_i - x_j\|^2 \geq (1 + \gamma \log n) \max\{b_\ell, b_{\ell'}\}\).

The authors of the paper are able to show that the distances between centroids are bounded from below given that the data is well-separated.

**Theorem 2.5** (Visualization theorem from [§3, [AHK18]]). Let \(X = \{x_1, x_2, \ldots, x_n\} \subset \mathbb{R}^d\) be \(\gamma\)-spherical and \(\gamma\)-well-separated clusterable data with \(C_1, C_2, \ldots, C_n\) defining the \(k\) individual clusters of size at least \(0.1(n/k)\), where \(k \ll n^{1/5}\). We choose \(\tau^2_\ell = \frac{1}{\gamma} \min_{j \in [n]} \{i : \|x_i - x_j\|^2 \} (\forall i \in [n]), h = 1, \) and any \(\alpha\) satisfying \(k^2 \sqrt{n} \log n \ll \alpha \ll n\).

Let \(Y^{(T)}\) be the output of t-SNE after \(T = \Phi\left(\frac{n \log n}{\alpha}\right)\) iterations on input \(X\) with the above parameters. Then with probability at least 0.99 over the choice of the initialization, \(Y^{(T)}\) is a full visualization of \(X\).

See the full paper [AHK18] for proof of the above theorem.

2.3. **Sampling.** As motivated in the introduction, using sampling techniques of Big data can be necessary to be able to deal with the sheer size of the data to be handled.
Perhaps the most obvious sampling technique is simply random sampling. Random sampling has the advantages of being very simple to implement and being unbiased. That means that in theory, random sampling produces a sample in which “different groups” or “types” of datapoints are represented in the same proportions as in the full dataset. While this might sound almost perfect, it is important to stress that this is true only in theory and that real-life results can be very far off from what we were hoping to see. Due to its nature, we have no control of the sampling process. Therefore, we can not guarantee that what we may think of being the constituent groups of data will be sampled proportionately.

An example of a deterministic sampling technique with a promising idea behind it is sampling according to hubness, i.e., the tendency of high-dimensional data to contain points (hubs) that frequently appear as being close to other points. More specifically, let $k$ be a positive integer, then the hubs are those points most frequently occurring in the lists of $k$ nearest neighbors of all other points (see Tomasev, et al. [TRMI14]). Hubness seem to be a good measure of “point centrality” and so the top $p$ percent of the dataset (ranked according to hubness) could be seen as cluster prototypes. However, there are cases where hubness gives outputs in which not every cluster is well-represented. For example, imagine a dataset containing two well-separated clusters out of which one contains significantly more datapoints than the other. In that case it would be easy to misrepresent the data by major hubs as they will almost certainly all be from the larger cluster. The same problem applies generally to situations of multiple clusters where only a small number of them make for the vast majority of the entire data. This is something that we want to solve by using random walk sampling.

Before going into random walk sampling, we give definitions of some of the more important notions of the theory behind random walks. We use the following notation: let $A^T$ be the transpose of matrix $A$ and let $\| \cdot \|_1$ be the $L^1$ norm.

**Definition 7.** Let $M$ be a finite state Markov chain and let $P$ denote the probability transition matrix of $M$. A vector $\pi$ satisfying the condition

\[
\pi = P^T \pi,
\]

is said to be a stationary distribution of $P$ if every entry is non-negative and it is normalized such that $\| \pi \|_1 = 1$. 
Remark. The stationary distribution as defined in Definition 7 is an eigenvector of eigenvalue 1 for $P^T$. The existence of the stationary distribution $\pi$ is given in the following lemma.

Lemma 2.6. Every transition matrix $P$ and its transpose have an eigenvector with corresponding eigenvalue 1.

Proof. For an $n \times n$ probability transition matrix $P$ whose row entries sum up to 1, $\sum_j P_{ij} = 1$, we see that by multiplying $P$ by the column vector $[1, 1, \ldots, 1]^T$ we get

$$
\begin{pmatrix}
p_{11} & p_{12} & \cdots & p_{1n} \\
p_{21} & p_{22} & \cdots & p_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
p_{n1} & p_{n2} & \cdots & p_{nn}
\end{pmatrix}
\begin{pmatrix}
1 \\
1 \\
\vdots \\
1
\end{pmatrix}
= 
\begin{pmatrix}
p_{11} + p_{12} + \cdots + p_{1n} \\
p_{21} + p_{22} + \cdots + p_{2n} \\
\vdots \\
p_{n1} + p_{n2} \cdots + p_{nn}
\end{pmatrix}
\begin{pmatrix}
1 \\
1 \\
\vdots \\
1
\end{pmatrix}
= 1 \cdot 
\begin{pmatrix}
1 \\
1 \\
\vdots \\
1
\end{pmatrix}.
$$

This comes from the property of probability transition matrices that its row entries always sum up to 1 and thus we have shown that every probability transition matrix $P$ has an eigenvector corresponding to eigenvalue 1. Since this holds for matrix $P$, it holds also for its transpose. See [Mey00] for the proof of the last statement. □

Random walk sampling can be thought of as a combination of the previously mentioned two sampling techniques, adding a certain level of randomness to the idea of sampling the “most representative points” in a dataset – so called “landmarks”. It is based on the theory of Markov chain random walks, specifically concerning the stationary distribution of a transition matrix. Entries $\pi_i$ of the stationary distribution can be thought of as the limit of the proportion of time that the process is in state $i$, when the number of “steps” in the Markov chain random walk goes to infinity. We believe that there is inherent value to sampling according to the stationary distribution $\pi$ since the “most visited” points in a dataset, i.e. $x_i \in X$ that correspond to the highest $\pi_i$ in the stationary distribution vector seem to be the real landmarks of a dataset, acting as true group-prototypes, regardless of the size of that group.

The problem, however, is that calculating the stationary distribution for a random walk on a large dataset of size $N$ is computationally infeasible as it includes finding roots to a polynomial of degree $N$. This is where approximating the sampling from the stationary distribution by means of random walk sampling becomes useful. It may be noted that Google’s PageRank algorithm is based upon the idea of approximating the stationary distribution $\pi$ of a graph (see [BL06]). Sampling
on graphs using random walk has been considered by Lovász in [Lov96]. An algorithm for random walk sampling of a point, as proposed in Basirian, Jung [BJ17] is: Select a starting point for the random walk (a seed) and perform a length-$L$ random walk according to the transition matrix $P$, taking then the last visited point and including it into the sample. By choosing a sufficiently large $L$, that is, in the limit $L \to \infty$ the sampled subset are the points in the stationary distribution with highest probability.

3. Methods

In this section we explain the procedure of our testing and what data is saved for analysis. The process is identical for each tested dataset.

3.1. Experimental setup. To reiterate: the research question that we will answer in this thesis is whether or not a random walk sampling method or using hubness for sampling gives better results than pure random sampling. Throughout the experiments, we have used three quality measures to see how successful each technique is, comparing the different sampling techniques with each other as well as with quality measure results obtained from running t-SNE on the entire dataset. Our three quality measures are trustworthiness, continuity and procrustes.

Definition 8. Analogous to [vdMPvdH09], we define here the trustworthiness of an embedding $Y = \{y_1, y_2, \ldots, y_N\} \subset \mathbb{R}^s$ of a high-dimensional dataset $X = \{x_1, x_2, \ldots, x_N\} \subset \mathbb{R}^d$ with $k$ degrees of freedom as

\[
T(k) = 1 - \frac{2}{Nk(2N - 3k - 1)} \sum_{i=1}^{N} \sum_{j \in U_{i}^{(k)}} (\max(0, r(i, j) - k)),
\]

where $r(i, j)$ represents the rank\(^3\) of the low-dimensional datapoint $y_j$ according to the pairwise distances between the low-dimensional datapoints. The set $U_{i}^{(k)}$ contains the points that are among the $k$ nearest neighbors to datapoint with index $i$ (denoted $x_i$ when in $\mathbb{R}^d$ and $y_i$ when in $\mathbb{R}^s$) in the low-dimensional space but not in the high-dimensional space.

Definition 9. Analogous to [vdMPvdH09], we define the continuity of an embedding $Y = \{y_1, y_2, \ldots, y_N\} \subset \mathbb{R}^s$ of a high-dimensional dataset $X = \{x_1, x_2, \ldots, x_N\} \subset \mathbb{R}^d$ as

\[
C(k) = \frac{\sum_{i=1}^{N} \sum_{j \in U_{i}^{(k)}} r(i, j) \cdot r_{ij}(k)}{\sum_{i=1}^{N} \sum_{j \in U_{i}^{(k)}} r_{ij}(k)}
\]

where $r_{ij}(k)$ is the rank of $y_j$ according to the pairwise distances between $y_i$ and its $k$ nearest neighbors.

\(^3\)By rank we mean the rank based on closeness to the point $y_i$; if $y_j$ is the nearest neighbor to $y_i$, then $r(i, j) = 1$. Similarly, if $y_i$ is the fifth nearest neighbor to $y_i$, then $r(i, j) = 5$.\)
$\mathbb{R}^d$ with $k$ degrees of freedom as

$$C(k) = 1 - \frac{2}{Nk(2N - 3k - 1)} \sum_{i=1}^{N} \sum_{j \in V_i^{(k)}} (\max(0, \hat{r}(i,j) - k)),$$

where $\hat{r}(i,j)$ represents the rank of the high-dimensional datapoint $x_j$ according to the pairwise distances between the high-dimensional datapoints. The set $V_i^{(k)}$ contains the points that are among the $k$ nearest neighbors to datapoint with index $i$ in the high-dimensional space but not in the low-dimensional space.

These two quality measures were used by van der Maaten, Postma, and van den Herik in their paper when comparing a large number of DR techniques [vdMPvdH09]. Also, like van der Maaten et al. in that paper, we are calculating trustworthiness and continuity with 12 degrees of freedom throughout the experiments. To summarize them, we can say that $T(k)$ is penalized if distant points become neighbors while $C(k)$ is penalized if neighboring points become distant.

Our “procrustes” quality measure is the mean squared error (MSE) of two low-dimensional embeddings that have been optimally “superimposed” onto each other. The process of full procrustes superimposition consists of optimally translating, rotating, uniformly scaling and even reflecting one of the embeddings so that the MSE is minimized. This is a measure of how similar the embedding of a sample is compared to the original embedding i.e. having run t-SNE on the entire dataset and then picked out the points that were chosen by a sampling method.

The first part of the experiment process for a specific dataset is to run the t-SNE algorithm on the entire dataset 10 times. The reason for that lies in the fact that t-SNE’s initial embedding is random so the local minimum that the gradient descent finds can vary between different runs. For each of the 10 embeddings of t-SNE applied to the full dataset, trustworthiness and continuity are calculated, and the time is kept for how long it took to run it.

Then, a sample size is chosen. We have tested the sampling techniques on sample sizes ranging from 10% to 50% in steps of 5%. A total of 4 sampling techniques are being tested: random sampling, two different random walk samplings and sampling based on hubness. The difference between the two random walk sampling methods is in the way how they decide which point to take the next step to. When having randomly chosen a seed datapoint, the random walk algorithm translates the distances between the seed and remaining datapoints into weights $w_{ij}$. The general requirement for the weights is that they sum up to 1, $\sum_{i \notin j}^{N} w_{ij} = 1$. 
and that closer neighbors have larger assigned weights than those further away, i.e. $\|x_i - x_j\| < \|x_i - x_k\| \implies w_{ij} > w_{ik}$. The first random walk sampling calculates those weights by taking the inverses of the distances between pairs of points and scaling them all by a constant that allows them to sum up to 1. To avoid computational problems with points having distance close to zero, we add a shrinkage weight of $\alpha = 1$ to both the numerator and denominator when taking the inverse distances. The second random walk sampling uses the affinities $p_{ij}$ for the high-dimensional data that t-SNE itself uses to do gradient descent. For convenience of notation, we will from now use the following abbreviations when referring to the four sampling methods: $rs$ for random sampling, $rw_1$ and $rw_2$ for the first and second random walk sampling methods respectively and $hs$ for hubness-based sampling.

For a fixed sampling size, each of the three sampling techniques that are (partially) random will make 30 samples and run t-SNE on each of them. For each of the runs, the embeddings, three quality measures as well as runtimes are saved. Hubness sampling, even though deterministic is also tested with 30 t-SNE runs on the top $p$-percent of hubs due to the random initialization for each run of t-SNE. Embeddings, quality measures and runtimes are all being saved, same as for the other sampling techniques.

After an embedding for a sample has been reached, the process of fitting the unsampled high-dimensional points into the low-dimensional space begins. Fitting the previously unsampled datapoints is done through $k$NN regression. The algorithm of $k$NN regression does the following for every unsampled $x_i \in \mathbb{R}^d$: it looks at the closest neighbors to $x_i$ in the high-dimensional space and picks the $k$ closest ones that have been sampled; then it transfers the distances to those points into weights in the same fashion as described for $rw_1$ and finally calculates the low-dimensional coordinates of $x_i$ as a weighted sum of the low-dimensional coordinates of its $k$ nearest neighbors from the high-dimensional space that were sampled. For more information about $k$NN regression, see [JWHT13]. After having fit all the points from the dataset into the low-dimensional space, quality measures, runtimes and the embeddings themselves are being saved for later analysis. In our case we have used $k$NN regression with parameter $k = 5$.

A Github repository containing all the .m files used for this thesis is available at the following link: Matlab programs used for this thesis. Alternatively, see [Bul19].
3.2. Datasets. The list of used datasets, along with their sizes and references are given in Table 1.

<table>
<thead>
<tr>
<th>dataset name</th>
<th>dataset size</th>
<th>dataset source reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>orl</td>
<td>400 × 396</td>
<td>[SH94]</td>
</tr>
<tr>
<td>seismic</td>
<td>646 × 24</td>
<td>[SW10]</td>
</tr>
<tr>
<td>har</td>
<td>735 × 561</td>
<td>[AGO+12]</td>
</tr>
<tr>
<td>svhn</td>
<td>732 × 1024</td>
<td>[NWC+11]</td>
</tr>
<tr>
<td>cnae9</td>
<td>1080 × 856</td>
<td>[CO09]</td>
</tr>
<tr>
<td>coil20</td>
<td>1440 × 400</td>
<td>[NNM+96]</td>
</tr>
<tr>
<td>secom</td>
<td>1576 × 590</td>
<td>[MM08]</td>
</tr>
<tr>
<td>bank</td>
<td>2059 × 63</td>
<td>[MCR14]</td>
</tr>
<tr>
<td>cifar10</td>
<td>3250 × 102</td>
<td>[KH09]</td>
</tr>
</tbody>
</table>

Table 1. Datasets used throughout our testing.

We give a short description of each dataset below.

(1) **orl**: Face images from 40 different subjects.
(2) **seismic**: Data used to forecast seismic bumps in a coal mine.
(3) **har**: Data from 30 subjects performing activities of daily living, used for human activity recognition.
(4) **svhn**: Street View House Numbers – Computer Vision dataset of images of digits 0 to 9 from Google Street View.
(5) **cnae9**: Free text descriptions of Brazilian companies in the National Classification of Economic Activities, split in 9 classes based on economic activity.
(6) **coil20**: Columbia University Image Library, consisting of images of 20 types of common objects.
(7) **secom**: Data from a semiconductor manufacturing process, used for training failure detectors.
(8) **bank**: Direct marketing campaign data of a Portuguese bank used to predict whether a client will subscribe to a banking product or not.
(9) **cifar10**: Standard Computer Vision research dataset consisting of images of animals and vehicles, used for training image classifiers.
In this section we present a chosen subset of the most interesting data obtained throughout the experiments. The reason for not including all the data is that there is simply too much raw data gathered to be presented in a concise way. To give the reader an idea about how much raw data was collected during the experiments, the total file sizes came to almost 10GB of memory.

We first take a look at the results before applying kNN regression. Since the main research question is only concerning the quality of sampling methods, it is needed to compare them before applying kNN regression as it will affect the quality of each embedding in a different way and it is hard to predict in which manner exactly.

Figures 2 and 3 show the average values for quality measures depending on the sample size.

![Figure 2. Trustworthiness depending on sampling size before applying kNN regression.](image)

The graphs showing trustworthiness and continuity for the 4 sampling techniques before applying kNN regression show very similar results. The general trend is that quality improves with sample size, but the rate of improvement slows down quickly. All sampling techniques have relatively similar results, but rw2 seems to be the only one to show a significant improvement over the others, when it is possible to decide a clear winner at all.
We accompany the results from these two graphs with a set of hypothesis tests whose results are available in Table 2 and 3. The hypothesis tests assure us that $rw_2$ gives significantly better results for trustworthiness and continuity than $rs$. In the first case, the null hypothesis is that trustworthiness values for $rs$ and $rw_2$ come from a distribution with the same mean. The alternative hypothesis is that the mean for the trustworthiness values corresponding to $rw_2$ is higher. By setting the confidence level to 99%, we reject the null hypothesis for any case where the T-score is greater than 2.46. Situations in which we reject the null hypothesis in favor of accepting that $rw_2$ gives better trustworthiness values than $rs$ are coloured blue for easier notice.

Similarly, in the second case, the null hypothesis is that continuity values for $rs$ and $rw_2$ come from a distribution with the same mean. The alternative hypothesis is that the mean for the continuity values corresponding to $rw_2$ is higher. By setting the confidence level to 99%, we reject the null hypothesis for any case where the T-score is greater than 2.46. Situations in which we reject the null hypothesis in favor of accepting that $rw_2$ gives better continuity values than $rs$ are coloured blue for easier notice.
Table 2. T-scores from hypothesis tests comparing trustworthiness of $rw_2$ and $rs$ before applying kNN regression.

<table>
<thead>
<tr>
<th>data</th>
<th>10%</th>
<th>15%</th>
<th>20%</th>
<th>25%</th>
<th>30%</th>
<th>35%</th>
<th>40%</th>
<th>45%</th>
<th>50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>orl</td>
<td>1.237</td>
<td>0.105</td>
<td>0.388</td>
<td>-1.457</td>
<td>-1.671</td>
<td>0.177</td>
<td>-0.779</td>
<td>-0.156</td>
<td>-1.293</td>
</tr>
<tr>
<td>seismic</td>
<td>3.796</td>
<td>1.901</td>
<td>-1.585</td>
<td>0.143</td>
<td>4.668</td>
<td>6.634</td>
<td>-1.225</td>
<td>11.339</td>
<td>-0.666</td>
</tr>
<tr>
<td>har</td>
<td>-2.079</td>
<td>-0.619</td>
<td>-1.819</td>
<td>0.989</td>
<td>2.292</td>
<td>2.979</td>
<td>5.137</td>
<td>8.203</td>
<td>10.97</td>
</tr>
<tr>
<td>secom</td>
<td>11.715</td>
<td>19.057</td>
<td>27.909</td>
<td>29.79</td>
<td>28.05</td>
<td>35.432</td>
<td>28.783</td>
<td>34.857</td>
<td></td>
</tr>
<tr>
<td>bank</td>
<td>10.5</td>
<td>17.511</td>
<td>17.073</td>
<td>22.307</td>
<td>30.983</td>
<td>19.734</td>
<td>23.281</td>
<td>30.087</td>
<td>34.627</td>
</tr>
<tr>
<td>cifar10</td>
<td>39.992</td>
<td>48.887</td>
<td>52.264</td>
<td>59.931</td>
<td>58.073</td>
<td>72.108</td>
<td>78.668</td>
<td>60.415</td>
<td>59.548</td>
</tr>
</tbody>
</table>

Table 3. T-scores from hypothesis tests comparing continuity of $rw_2$ and $rs$ before applying kNN regression.

<table>
<thead>
<tr>
<th>data</th>
<th>10%</th>
<th>15%</th>
<th>20%</th>
<th>25%</th>
<th>30%</th>
<th>35%</th>
<th>40%</th>
<th>45%</th>
<th>50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>orl</td>
<td>1.819</td>
<td>-0.311</td>
<td>0.356</td>
<td>-0.951</td>
<td>-1.558</td>
<td>-0.333</td>
<td>-0.671</td>
<td>-0.356</td>
<td>-0.671</td>
</tr>
<tr>
<td>seismic</td>
<td>2.888</td>
<td>0.701</td>
<td>-1.765</td>
<td>0.18</td>
<td>4.478</td>
<td>6.211</td>
<td>-1.243</td>
<td>9.311</td>
<td>-0.738</td>
</tr>
<tr>
<td>har</td>
<td>-2.236</td>
<td>-1.138</td>
<td>-4.361</td>
<td>-1.384</td>
<td>-1.141</td>
<td>-0.072</td>
<td>-0.372</td>
<td>-0.105</td>
<td>2.162</td>
</tr>
<tr>
<td>cnae9</td>
<td>17.154</td>
<td>19.421</td>
<td>23.989</td>
<td>27.284</td>
<td>27.076</td>
<td>33.238</td>
<td>37.471</td>
<td>26.29</td>
<td>32.58</td>
</tr>
<tr>
<td>coil20</td>
<td>15.021</td>
<td>13.697</td>
<td>18.044</td>
<td>17.402</td>
<td>18.665</td>
<td>17.7</td>
<td>16.415</td>
<td>15.551</td>
<td>15.096</td>
</tr>
<tr>
<td>cifar10</td>
<td>38.92</td>
<td>39.147</td>
<td>37.811</td>
<td>48.827</td>
<td>30.895</td>
<td>30.725</td>
<td>41.455</td>
<td>45.367</td>
<td>32.105</td>
</tr>
</tbody>
</table>

The results vary slightly across datasets, the larger ones showing a clearer preference for $rw_2$ rather than $rs$. This is very good because our target is to optimize performance for large datasets.

Figures 4, 5, 6 are depicting the trends of trustworthiness, continuity and procrustes for different datasets, depending on the sample sizes, post-kNN. The fact that these measures have been calculated after performing kNN regression might explain the quality loss compared to the results of before applying kNN regression. We believe that the issue is that kNN regression using weights as for the first random walk is simply not a good enough way to estimate the low-dimensional
coordinates of the previously unsampled points. More precisely, it does not replicate the effect of applying t-SNE on the entire dataset in the way that it clearly separates clusters and compresses preserved clusters into dense, clearly identifiable groups.

For reference, Table 4 shows the average scores for trustworthiness and continuity obtained when running t-SNE on the complete datasets 10 times. Looking at these results, we can happily conclude that in many cases our 4 sampling methods, especially $rw_2$ which was the most distinguished among them gave quite similar quality measure results compared to those of applying t-SNE on the whole dataset.

<table>
<thead>
<tr>
<th>dataset</th>
<th>trustworthiness</th>
<th>continuity</th>
</tr>
</thead>
<tbody>
<tr>
<td>orl</td>
<td>0.5364</td>
<td>0.5338</td>
</tr>
<tr>
<td>seismic</td>
<td>0.98648</td>
<td>0.98581</td>
</tr>
<tr>
<td>har</td>
<td>0.9557</td>
<td>0.95456</td>
</tr>
<tr>
<td>svhn</td>
<td>0.90234</td>
<td>0.92966</td>
</tr>
<tr>
<td>cnae9</td>
<td>0.78753</td>
<td>0.91551</td>
</tr>
<tr>
<td>coil20</td>
<td>0.99199</td>
<td>0.98928</td>
</tr>
<tr>
<td>secom</td>
<td>0.92857</td>
<td>0.94252</td>
</tr>
<tr>
<td>bank</td>
<td>0.96057</td>
<td>0.91299</td>
</tr>
<tr>
<td>cifar10</td>
<td>0.88988</td>
<td>0.92609</td>
</tr>
</tbody>
</table>

Table 4. Average trustworthiness and continuity scores from running t-SNE on the entire datasets.

The graphs for trustworthiness and continuity after $k$NN regression clearly show that while embeddings of all 4 sampling techniques suffer in quality after applying $k$NN regression on the sampled results of t-SNE, sampling according to hubness is the only one that manages to separate itself from the others in terms of it not suffering as much as the rest. As the results shown in the graphs are quite clear, we believe that a separate set of hypothesis tests is not needed here.

We notice a somewhat counter-intuitive feature in the results, namely that trustworthiness and continuity scores tend to drop as the sample size increases. One idea as to why this is happening would be as follows. Trustworthiness and continuity are measures that are “concerned” about keeping the $k$ nearest neighbors from the high-dimensionality space preserved when doing DR. Since the $k$NN regression
Figure 4. Trustworthiness depending on sampling size after applying $k$NN regression.

Figure 5. Continuity depending on sampling size after applying $k$NN regression.
algorithm adds unsampled points close to their \(k\)NN points (which is exactly what gives high scores for trustworthiness and continuity), then the cases with smaller sample sizes — thus leaving larger portions of the data for \(k\)NN regression — will score better than those of big samples.

On the other hand, procrustes scores after \(k\)NN regression vary greatly and it is hard to say with confidence that one sampling technique is clearly better than the remaining ones. Our estimate is that the reason for that lies in the nature of t-SNE itself. Due to the way it calculates the pairwise affinities \(p_{ij}, q_{ij}\) for the high- and low-dimensional space, its primary focus is to preserve local information and not the general relative positioning of clusters to each other. That is why procrustes, which is only able to find the best superpositioning of two embeddings using linear transformations suffers in those cases where clusters are differently positioned in the low dimensional space compared to the embeddings from t-SNE applied to the full dataset — for it is the embeddings from t-SNE on the entire dataset that we are comparing the embeddings of combining sampling and \(k\)NN regression to.

**Figure 6.** Procrustes depending on sampling size after applying \(k\)NN regression.

**Figure 7** shows the average runtimes (in seconds) when applying each of the 4 sampling techniques and running t-SNE on the sample versus the runtimes of running t-SNE on the entire dataset. Unfortunately, the runtimes kept after applying
kNN regression to the t-SNE embedding of the sample were lost while running the experiments. However, recalling the times that were being displayed in Matlab’s command window during the experiments, we approximate the runtimes of KNN regression to be not more than 20% of the time it takes to sample a dataset and apply t-SNE on the sample.

There is a clear trend of datasets with more instances taking longer to run said trials on. It is worth to note that the runtimes of experiments consisting of sampling a dataset, applying t-SNE on the sample and even applying kNN regression on that result (again, approximating the time necessary to do kNN regression) is less than the average runtime for t-SNE on the entire dataset. The differences are in fact more pronounced for larger datasets so we can imagine that the time-savings using our procedure for much larger dataset will be significant enough for researchers looking to save time when doing DR techniques. The combination of time-savings with the average quality measures shown in Figure 2 and 3 seems as an attractive alternative to the lengthy runs of t-SNE on the unsampled dataset.

Finally we show an example of running t-SNE on the “coil20” dataset first without any sampling being used (see Figure 8), then comparing the embeddings after
using the 4 sampling techniques at a sampling rate of 25% (see Figure 9) and finally the results of applying kNN regression to the different samples of the dataset (see Figure 10).

The reason for choosing this particular dataset for visualization is that it nicely shows the main features of each of the sampling techniques that were discussed earlier in the paper. On one side we have random sampling which is taking equal proportions of each of the clusters into the sample. Opposite of that is hubness sampling which, as mentioned earlier, cares about the most central datapoints, preferring those coming from the largest clusters. The two random walk sampling methods show truly that their main features are a combination of those of naive random sampling and hubness sampling. In that regard rw2 is closer to hubness sampling than the first one, the reason lying in the way that its weights set is calculated, being more biased towards the closer neighbors, giving them even larger weights than rw1.

On the other hand, hubness sampling’s bias favoring only the largest clusters shows here why it had an upper hand compared to the other sampling techniques after applying kNN regression.

Figure 8. The result of t-SNE applied to the entire coil20 dataset.
Figure 9. The result of t-SNE applied to the 25% samples of the coil20 dataset.

Figure 10. The result of t-SNE applied to the 25% samples of the coil20 dataset in combination with $k$NN regression for fitting the previously unsampled datapoints.
5. Discussion and future work

The purpose of this thesis is twofold. Firstly we give an introduction to the use of DR techniques in visualization and prove that t-SNE in particular is successful at preserving well-separated clusters. Secondly, our thorough testing gives us the ability to draw conclusions about the quality of embeddings from running t-SNE on samples of data using different sampling techniques.

The main takeaway is that choice of sampling method, just as much as the choice of DR technique depends heavily on the dataset that we want to study and on the aspect of the dataset that we want to focus most on. Thus if one plans to apply PCA on a multidimensional dataset in order to examine its variance, and would like to combine it with one of our sampling techniques, then our suggestion would be to use random sampling. If on the other hand one wants to study the clusters of a dataset, then t-SNE paired with \( rw_2 \) or even hubness sampling is a good choice, depending on how much one cares about the preserving of smaller clusters.

We can intuitively explain why random walk sampling and hubness based sampling are a good choice when picking the most important points from a cluster in the high-dimensional space by the following toy example. Imagine that a cluster in \( \mathbb{R}^d \) consists only of a small number of points (say \( N \approx 5 \)) that lie very close to their least-squares line. Random walk sampling and hubness sampling will then pick out the points that are in the middle, which are arguably the most important points to sample to achieve cluster preserving, as these points best represent the cluster, in that they are closest to each other point on average.

We have statistically shown that when it comes to calculating trustworthiness and continuity of the embedding of t-SNE applied to the sampled data, \( rw_2 \) is better than pure random sampling. In the 90 hypothesis tests regarding results for trustworthiness, we have accepted the hypothesis of the second random walk sample being better 68 times with a significance level of 1%. It is worth noting that in the remaining 22 hypothesis tests where we could not reject the null hypothesis of the two sampling methods giving equally good results, we would not be able to accept the other alternative hypothesis of \( rs \) being better than \( rw_2 \) even once at that significance level. The results of hypothesis tests for continuity were similar: we accepted the alternative hypothesis of \( rw_2 \) being better 63 out of 90 times at a significance level of 1%, noting that should the alternative hypothesis been that \( rs \) is better, it would only be accepted once at the same significance level.
When it comes to future work on this topic, a direction for research would be to try out even more random walk samplings that would calculate their weights differently to the way our two random walks did. Another major point of improvement is to try and optimize the regression method for fitting previously unsampled points into the low-dimensional embedding. Here we have only used kNN regression with \( k = 5 \) and calculating the weights for the regression in the fashion of scaled inverse distance to the closest neighbors – equivalent to the way \( rw_1 \) calculates its weights. Perhaps another way of calculating those weights would be beneficial or even changing the parameter of \( k \) for the number of nearest neighbors observed.

Also, it would be very interesting to run our set of experiments for much larger datasets to see if the trade-off of losing some quality for saving time would seem even more attractive than now. As a matter of fact, our original goal with the thesis’ experiments was to try out 10 more datasets in addition to the ones analyzed now. Unfortunately, the total runtimes for experiments on datasets larger than those used here were too long, presenting a large performance problem for the specific hardware being used to do these calculations. For reference, the process of running our experiments on the largest of the presented datasets, “cifar10” took approximately 16 hours to run on a Microsoft Surface Book 2 with an Intel Core i7-8650U (4.2GHz), 16GB of RAM and a mobile NVIDIA GTX 1050 (2GB) graphics card. Due to the fact that t-SNE has a quadratic order of complexity \( O(N^2) \), the estimated runtimes for much larger datasets would be infeasibly long.
OPTIMIZING T-SNE USING RANDOM SAMPLING TECHNIQUES

REFERENCES


Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.


Nenad Tomasev, Milos Radovanovic, Dunja Mladenic, and Mirjana Ivanovic. The role of hubness in clustering high-dimensional data. IEEE Transactions on Knowledge and Data Engineering, 26(3):739–751, 2014.

