TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

ALEKSI HIETANEN
UNTITLED
Master's thesis

Examiner: John Doe

The examiner and topic of the thesis were approved on 11 September 2017

ABSTRACT

ALEKSI HIETANEN: Untitled
Tampere University of Technology
Master of Science Thesis, 50 pages
October 2017
Master's Degree Programme in Science and Engineering
Major: Theoretical computer science
Examiner: John Doe

Keywords: Key, Word, List

TODO

TIIVISTELMÄ

ALEKSI HIETANEN: Otsikoimaton
Tampereen teknillinen yliopisto
Diplomityö, 50 sivua
Lokakuu 2017
Teknis-luonnontieteellinen diplomi-insinöörin tutkinto-ohjelma
Pääaine: Ohjelmistotiede
Tarkastaja:

Avainsanat:

TODO

PREFACE

TODO

In Helsinki, Finland, on 31 May 2017

Aleksi Hietanen

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

CC license	Creative Commons license
LaTeX	typesetting system for scientific writing
SI system	Système international d'unités, International System of Units
TUT	Tampere University of Technology
URL	Uniform Resource Locator

a	acceleration
$\mathbf{F}$	force
m	mass

1. INTRODUCTION

Dimensionality reduction is one of the key research areas in the field of machine learning. As real world data sets are usually of high dimensionality, such as ..., methods to reduce this dimensionality are sought to alleviate the problems arising when dealing with high dimensional data.

At present, much of the data being produced, gathered and analyzed is very high dimensional. As a consequence, several challenges arise when analyses are being performed on these vast amounts of high dimensional data. At present,

Several problems arise when dealing with high dimensional data: - distances between pairs of points become less distinguished as dimensionality of the space increases [2]. - The volume of a space grows exponentially along with the number of dimensions. This leads to even large data sets appearing sparse in the feature space. Efficient sampling requires a lot of points - algorithms depend on dimensionality of the data being processed and large computational costs are accrued, especially sampling becomes - need for more training data, k-means example [56, p. 263] - algorithm complexity depends on the dimensionality of data being processed, large costs are accrued - from a more practical perspective, high dimensional data by itself is hard to directly reason about and difficult to visualize in any meaningful sense - In literature, problems arising from high dimensionality are collectively referred to as the *curse of dimensionality*, as first coined by Richard Bellman in the context of his work on dynamic programming [6]. To tackle these problems associated with high dimensional data, reduction of the dimensionality of data is sought through various means of feature selection and feature extraction. Feature selection is ..., whereas feature extraction ... . In this work we focus on purely unsupervised feature extraction through

Several methods for dimensionality reduction have been developed and attempting to cover a majority of them would be greatly out of scope of the work for this thesis. For comprehensive reviews on other prominent methods the reader is referred to [44, 8].

Fortunately, the so called manifold hypothesis [18] often holds for real world data sets. The manifold hypothesis posits that the high dimensional data in question lies near a much lower dimensional manifold. As an example, take This in turn implies that learning

This, together with the classical Johnson-Lindenstrauss lemma [31], which gives bounds to the error

The manifold assumption and manifold learning. In general, dimensionality reduction aims to preserve as much information of the underlying data as possible. Several different methods for dimensionality reduction have been developed over the years. Arguably the

most well known and applied method in practice is Principal Component Analysis (PCA), developed independently by ... [20] and ... [28] for the study of ... . Although very efficient to compute and easy to reason about, PCA however suffers from a few drawbacks, making it not ideal for data visualization purposes. the fact that it only aims to maximize the variance... thus local structure of the data is poorly preserved. On the other hand, methods that aim to preserve local structure of data typically suffer from the curse of dimensionality in several ways, such as ... .

Dimensionality reduction is a tool commonly used in disciplines such as ... for visualization of high dimensional data as 2 or 3 dimensional scatter plots.

In exploratory data analysis, visualization of the characteristics of data sets plays a key role.

In this work a method for improving the applicability of parametric t-SNE is proposed. In the proposed method a generative model in the form of a Variational Autoencoder is first trained and subsequently used to produce training data for the parametric t-SNE embedding. The benefits of this procedure are severalfold: . Further, each of these benefits are experimentally demonstrated on a variety of data sets, both quantitatively and qualitatively.

The structure of this thesis is as follows. In Chapter 2 the relevant background and related work to the contribution of this thesis is covered to give context to an audience not previously familiar with variational autoencoders or the t-SNE method for dimensionality reduction. The chapter is organized into two main sections, one that builds up to the theory and recent advances in variational autoencoders from the introduction of neural networks, and the other section presents the development of t-SNE up to its parametrization with neural networks.

The method developed for this thesis is presented in Chapter 3. The proposed learning and inference algorithms are presented, together with discussion on extensions and further variations. In addition, relevant implementation details for the application built to produce the results for the experiments chapter are briefly covered.

In Chapter 4 the proposed method is tested experimentally for different properties presented in the previous chapter, as well as compared with existing methods. The experiments chapter includes descriptions for the various data sets being used and definitions for the metrics employed to quantitatively assess the quality of the produced low dimensional embeddings by our method and the methods under comparison.

The final chapter presents discussion and conclusions for the work carried out in this thesis. Additionally, directions for future work are suggested.

2. BACKGROUND AND RELATED WORK

2.1 Variational Autoencoders

Before moving on to specialized artificial neural network structures and learning objectives.

2.1.1 Artificial Neural Networks

Artificial neural networks, or as commonly referred to simply as neural networks, are at the core of many recent advances in machine learning. Particularly, deep neural network architectures now hold many state-of-the-art results in fields including computer vision, natural language processing and speech recognition [37]. More broadly, neural networks are applicable

Supervised learning, [15].

Unsupervised and semi-supervised learning, representation learning.

Reinforcement learning, atari [46], go [57] and later go *tabula rasa* [58].

The development of ever faster hardware for accelerating the computations needed for deep learning has played a pivotal role in the resurgence of

A more simplistic view of neural networks would be to consider them as simply very powerful function approximators. The theoretical underpinning for this view is the universal approximation theorem [27].

Basic Principles
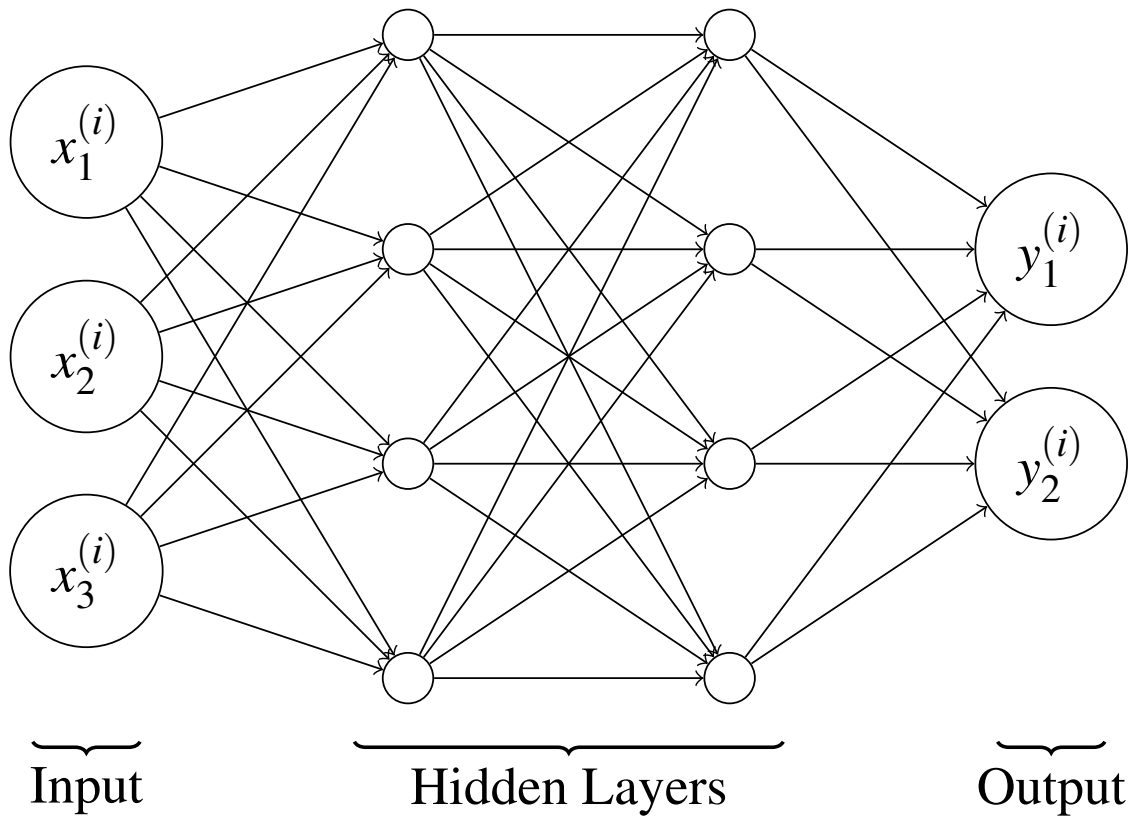
Figure 2.1. *Feed-forward neural network with input dimension 3, output dimension 2 and 2 hidden layers of dimension 4 each.*

Learning in Artificial Neural Networks

History of backprop (citations like/leading to [54]). SGD, Momentum, ... second order methods?

Modern Advances and Best Practices

2.1.2 Autoencoders
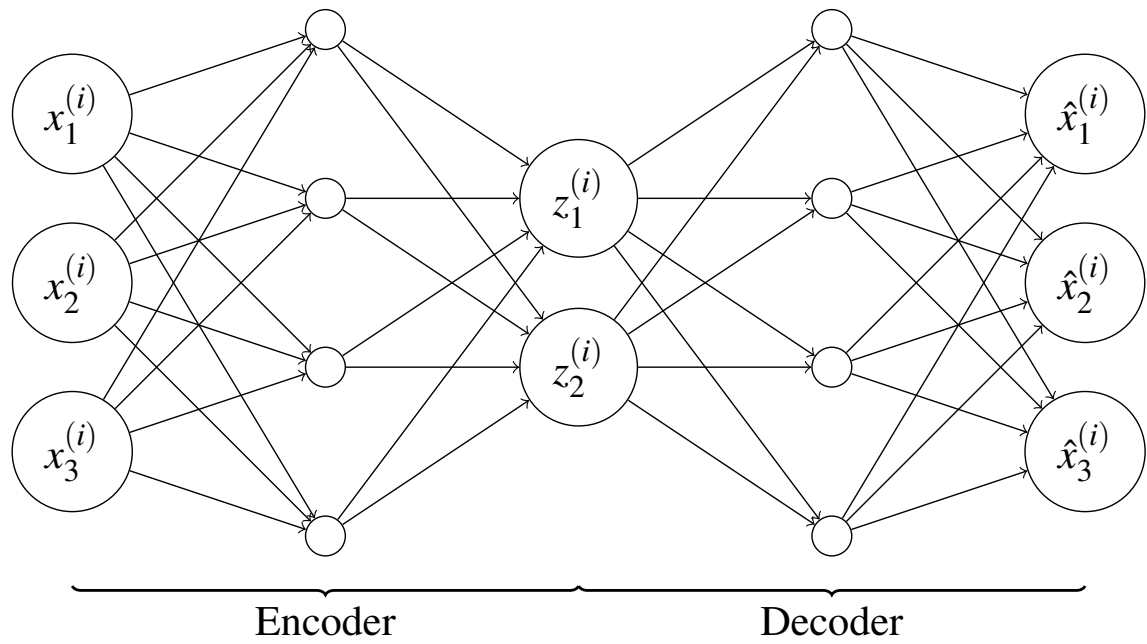
Figure 2.2. *Autoencoder*

2.1.3 Variational Bayesian Methods

Strategies for computing these intractable integrals

Further, it is possible to combine automatic differentiation with variational inference to further ease statistical model building and computational efficiency [36].
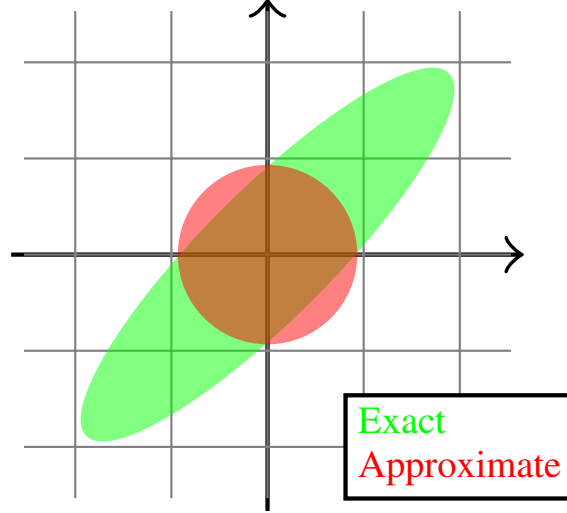
Figure 2.3. *Illustration of the mean-field approximation of a full rank Gaussian. Adapted from [7], Figure 1.*

Kullback-Leibler Divergence

The Kullback-Leibler divergence, or KL-divergence for short, is a measure of dissimilarity between two probability distributions.

For continuous probability distributions p and q the Kullback-Leibler divergence is defined as

$$D_{\mathrm{KL}}(p \parallel q) = \int p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})} d\mathbf{x},$$

and likewise for discrete probability distributions as

$$D_{\mathrm{KL}}(p \parallel q) = \sum_i p(i) \log \frac{p(i)}{q(i)}.$$

The following properties hold for

One must note however that KL-divergence is asymmetric, i.e. $D_{\mathrm{KL}}(p \parallel q) \neq D_{\mathrm{KL}}(q \parallel p)$, and thus cannot readily be used as a metric.

The Evidence Lower Bound

2.1.4   Auto Encoding Variational Bayes
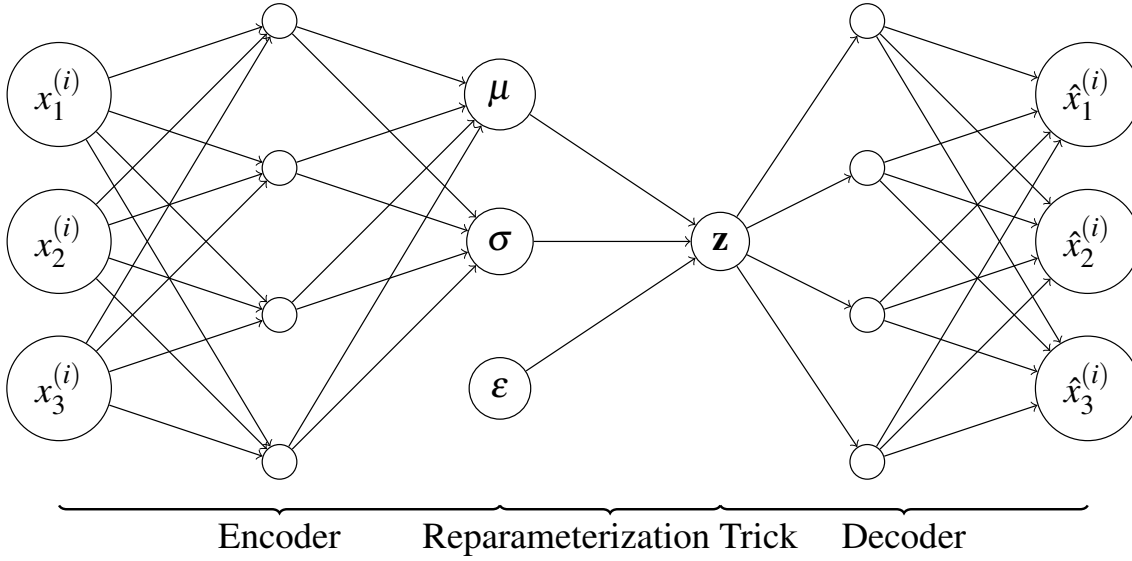
Figure 2.4. *Variational Autoencoder*
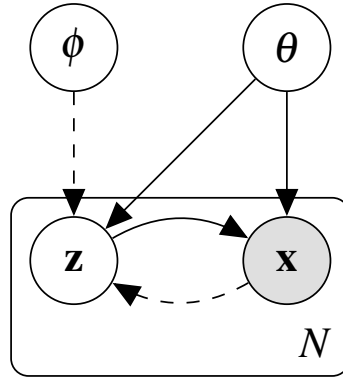
Figure 2.5. *The VAE graphical model, in plate notation. Adapted from [35], Figure 1.*

Derivation of the VAE Objective Function

The objective of VAEs is to model the data as well as possible, i.e. maximize the marginal likelihood $p_\theta(\mathbf{x}^{(i)})$ for each data point i. By an application of Jensen's inequality we can derive the following lower bound for the marginal likelihood under our latent variable

model:

$$\begin{aligned}
\log p_\theta(\mathbf{x}^{(i)}) &= \log \int p_\theta(\mathbf{x}^{(i)}|\mathbf{z})p(\mathbf{z})d\mathbf{z} \\
&= \log \int \frac{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}p_\theta(\mathbf{x}^{(i)}|\mathbf{z})p(\mathbf{z})d\mathbf{z} \\
&= \log \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\left[\frac{p_\theta(\mathbf{x}^{(i)}|\mathbf{z})p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\right] \\
&\geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\left[\log \frac{p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\right] + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\left[\log p_\theta(\mathbf{x}^{(i)}|\mathbf{z})\right] \\
&= -D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)}) \,\|\, p(\mathbf{z})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\left[\log p_\theta(\mathbf{x}^{(i)}|\mathbf{z})\right].
\end{aligned} \tag{2.1}$$

This bound is better known as the *evidence lower bound*, or ELBO for short. An interpretation for the terms of the last equation is that the objective is to have the posterior $q_\phi(\mathbf{z}|\mathbf{x}^{(i)})$ match the prior $p(\mathbf{z})$ as accurately as

In many cases the KL-divergence can be integrated analytically, for instance in the case of two Gaussian distributions. In cases where the KL term cannot be solved analytically, Monte Carlo methods are used to approximate the expectation.

Several modifications to this objective have been proposed, e.g. the β-VAE [25, 9], β-TCVAE [12] and InfoVAE [69].

Obtaining Differentiable Monte Carlo Estimates

To optimize the parameters θ and ϕ of our neural network using backpropagation we require a way to compute the gradient of the expectations of random variables. By introducing a random noise variable as input we are able to obtain samples from the latent distribution by using the *reparameterization trick*.

$$\widetilde{\mathbf{z}} = g_\phi(\varepsilon, \mathbf{z})$$

2.2 t-Distributed Stochastic Neighbor Embedding

Taking a slight departure from the preceding discussion on neural networks, the theory, and development leading to the acclaimed dimensionality reduction algorithm t-SNE will be presented. At the end of this section, a way to utilize the t-SNE objective to train an embedding function parameterized by a neural network will be shown, thereby connecting this section with the previous in a natural way, as well as covering the final preliminaries to the contribution of this thesis.

2.2.1 SNE

The Stochastic Neighbor Embedding (SNE) algorithm [26], developed by Hinton and Roweis in 2003, serves as the precursor to the later improved t-SNE algorithm. As such, SNE shares many similarities in its construction with t-SNE, and thus by first developing an understanding of SNE and its pitfalls the motivation behind the t-SNE algorithm become more apparent.

As the algorithm's name suggests, the approach SNE takes to perform dimensionality reduction is probabilistic in nature. Without delving directly into the mathematical details, an intuitive overview of the algorithm is that it assigns a probability to the event that each pair of points gets picked in the input space, after which the distribution of points in the output space is attempted to be matched with that of the input space under a similar probability measure. The natural way to match two distributions is familiar from the discussion on variational bayes methods, namely the KL-divergence between these two is used as a loss function for the embedding of data points, which can be optimized with various methods, including the previously covered gradient descent.

To formalize these notions, we begin with how the conditional probability distribution $p_{j|i}$ in the input space is defined by the equation

$$p_{j|i} = \frac{exp(-\left\|\mathbf{x}_i - \mathbf{x}_j\right\|^2 / 2\sigma_i^2)}{\sum_{k \neq i} exp(-\left\|\mathbf{x}_i - \mathbf{x}_k\right\|^2 / 2\sigma_i^2)},$$

where σ_i is determined separately for each data point i by forcing the perplexity of P_i equal to some preset value. Perplexity is an information theoretic measure of how well a distribution predicts its samples. For a given probability distribution, perplexity is defined as the exponentiation of the distribution's Shannon entropy, e.g. in the case of our discrete P_i:

$$b^{H(P_i)} = b^{-\sum_{j=1}^n p_{j|i} \log_b p_{j|i}},$$

where b is the unit used to measure entropy in, typically chosen to be either 2 or e, for bits or nats, respectively. From the definition we can see that larger σ will be chosen in low

density areas and smaller σ will be used in high density areas. Effectively, σ is adapted to fit the core assumption that the data points lie on a uniformly sampled manifold within the high dimensional ambient manifold $\mathbb{R}^d$. Perplexity is thus a value proportional to the number of effective local neighbors considered by the algorithm when the embedding is being optimized.

The conditional distribution in the output space is similarly defined with a Gaussian kernel:

$$q_{j|i} = \frac{exp(-\|\mathbf{y}_i - \mathbf{y}_j\|^2)}{\sum_{k \neq i} exp(-\|\mathbf{y}_i - \mathbf{y}_k\|^2)},$$

where each $\mathbf{y}_i$ represents the point in the embedding for data point $\mathbf{x}_i$. A notable difference is the absence of the variance term $2\sigma_i^2$, which can be interpreted as having been set to 1 throughout.

Now that we have defined both the output and input distributions for points, the objective function can be formulated as the sum of KL-divergences between each input-output distribution pair

$$C = \sum_i D_{\text{KL}}(P_i \parallel Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}. \tag{2.2}$$

An important fact to note is that the objective function is in fact non-convex, which implies the unfortunate property that the optimization of the objective function is not guaranteed to terminate at the global optimum due to the potential existence of multiple local minima. With random restarts and other optimization methods, such as gradient descent with momentum, it is possible to attempt to escape poor local minima.

Another important observation to make is how the non-symmetry of the KL-divergence plays a role in penalizing the embedding. [Note: Explain how.]

Gradient descent is typically used to optimize the embedding. The partial derivatives w.r.t. each point in the embedding is given by

$$\frac{\partial C}{\partial \mathbf{y}_i} = 2 \sum_j (\mathbf{y}_i - \mathbf{y}_j)(p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j}) \tag{2.3}$$

This has the intuitive interpretation of a force directed layouting algorithm, where each pair of points is either pushing or pulling on each other, depending on whether they are less or more likely to be neighbors under the output's conditional distribution than the input's.

2.2.2   t-SNE

The t-SNE algorithm aims to address two issues of the original SNE algorithm. First, through considering joint distributions, instead of conditional distributions, the form of the gradient of the objective function becomes easier to compute. Second, one of the main issues with plain SNE is the so called *crowding problem*, which is addressed by considering a heavier tailed probability distribution function in the output space, namely Student's t-Distribution. Together these improvements enable better separation of natural clusters in the output space of the algorithm, as compared to the original SNE method.

Symmetric SNE [13] aims to provide an objective function whose gradient is easier to compute than the one of Equation 2.3.

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$$

$$q_{ij} = \frac{(1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2)^{-1}}{\sum_{k \neq l}(1 + \|\mathbf{y}_k - \mathbf{y}_l\|^2)^{-1}}$$

The t-SNE objective function is defined similarly to the SNE objective (cf. Equation 2.2), except the sum is taken over KL-divergences of the joint probabilities:

$$C_{\text{t-SNE}} = \sum_i D_{\text{KL}}(P_i \parallel Q_i) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}.$$

For which the following gradient can be derived, w.r.t. each $\mathbf{y}_i$:

$$\frac{\partial C_{\text{t-SNE}}}{\partial \mathbf{y}_i} =$$

To alleviate the crowding problem,

Methods to improve the runtime of t-SNE have been subsequently proposed: [68, 41]. The use of the fast multipole method [53] has not been explored due to no suitable expansion for the t-Distributed forces in the output space exist.

Worth a mention is the Stochastic Triplet Embedding [62] and Multiple Maps t-SNE [43] methods developed on top of t-SNE, developed to handle non-metric similarities present in the data.

2.2.3 Parametric t-SNE

t-SNE does not directly enable out of sample extension, but random-walk t-SNE has been developed for this purpose. Although the following pitfalls are present in this method: ...

A natural extension to t-SNE would be to learn a function parameterizing the embedding from input space to output space.

3. METHOD

3.1   Learning a Parametric Embedding Using VAE Sampling

3.2   Robustness to Sparse and Noisy Data

3.2   Robustness to Sparse and Noisy Data

3.3   Sampling from Hidden Layers

3.4   Inference with the Generative Model

3.5 Implementation

An accompanying implementation of the method has been made available at https://github.
com/ahie/vptsne. The implementation is built on top of Tensorflow [1] and takes advantage
of the Tensorflow Distributions [16] library. To take full advantage of the computation
graph architecture of Tensorflow and the paralellism provided by GPGPUs, both the t-
SNE loss and its gradient computation were implemented as custom CUDA [48] kernel
operations.

[Note: Describe that it also includes fast parallel implementation of NN-Descent [17]?]

4. EXPERIMENTS

The code and data used to reproduce all the results presented in this section have been made available at https://github.com/ahie/vptsne-results.

4.1 Data Sets

To validate the proposed method we studied the properties of the combined model on the four different data sets described below.

4.1.1 MNIST

The MNIST data set [38] contains labeled images of handwritten digits. Each image contains a single digit in gray-scale with a resolution of 28x28 pixels. The data set is split into 60000 training images and 10000 test images. All networks were trained solely on the training data set, while the remaining test examples were reserved for assessing the capability of the network to generalize to out-of-sample extensions. In the unsupervised setting considered here, the provided labels were only used for visualizing the embeddings and evaluating the embedding quality of out-of-sample extensions via a 1-NN classifier.

4.1.2 Fashion-MNIST

The Fashion-MNIST data set [67] was designed as a drop-in replacement for the MNIST data set and thus contains the same number of grayscale training and test images of equal dimensions. The images themselves consist of 10 different classes of fashion items, such as T-shirts, trousers, sneakers, and handbags. The motivation for considering this data set in addition to MNIST is the fact that it has been shown to be much harder to learn even with supervised computer vision methods.

4.1.3 Mass Cytometry

Cytometry is the measurement of biological characteristics of cells. In mass cytometry, time-of-flight mass spectrometry is used to measure the counts of cellular proteins present within a single cell by tagging proteins with their corresponding antibodies that have been conjugated with specific heavy metals.

As a comparatively new methodological development, mass cytometry enables simultaneous measurement of a considerably greater number of features compared to fluorescence-based flow cytometry. A greater number of features, however, poses new challenges in

analyzing the measurement results. To aid the analysis of this high dimensional data it is common to visualize the data as a 2-D or 3-D scatter plot using t-SNE, due to its capability to separate biologically relevant subpopulations of cells in the produced embedding [4]. However, as a high-throughput method, mass cytometry data sets under analysis can grow to be millions of data points in size, prompting the use of more scalable algorithms. For a more thorough review of mass cytometry and associated analytical challenges, the reader is referred to [59].

In our experiments, we consider two different mass cytometry data sets. As the first data set, henceforth referred to as Cytometry A, we use the publicly available data of [39], which has been used to demonstrate the effectiveness of the Phenograph clustering algorithm on mass cytometry data. The data set consists of 81000 data points of 13 dimensions corresponding to the normalized counts of cell surface proteins belonging to distinct clusters of differentiation. As a benchmark data set for the Phenograph clustering algorithm, we additionally include the labels produced by Phenograph in our analysis.

The second mass cytometry data set (Cytometry B) contains data gathered from ovarian cancer patients at different phases of treatment. The number of points in the entire data set is 1.4 million, each of which has 23 dimensions.

4.2   Evaluation Metrics

To quantitatively evaluate the quality of embeddings two different measures are used in this work. By quality of the embeddings we are interested in how well the local structure of the data is preserved. The choice of the two metrics used in this work follows those used in the original parametric t-SNE paper [40]. A short description for each is given below.

4.2.1   k-Nearest Neighbor Classifier

Given a labeled data set a k-nearest neighbor (k-NN) classifier can be used to roughly assess how well different classes are separated in the output space of an embedding. A k-NN classifier works simply by returning the majority label of a given point's k-nearest neighbors in the training data set. For example in the 1-NN classifier case, the decision boundary can be geometrically visualized as a Voronoi diagram as in Figure 4.1.

Throughout the experiments, the performance of nearest neighbor classifiers in the output space of embeddings will be evaluated. More precisely, the accuracy of a 1-nearest neighbor classifier with the Euclidean metric for determining neighbor distance is used.

For parametric methods, the classifier can be trained on the embedding of training data points and separately tested on the training set. This enables evaluating the ability of the embedding function to generalize to unseen data. On the other hand, for non-parametric methods, the final embedded data points need to be split into train and test sets for the 1-NN classifier.
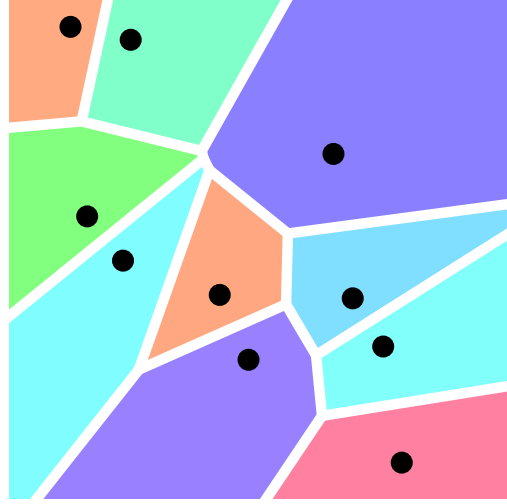
Figure 4.1. *Voronoi diagram illustrating the decision boundaries of 10 different classes of a 1-NN classifier. Black dots represent the training data points, and the different colors correspond to the label of the data point for that region. Any point to be classified would receive the label corresponding to the location in the tesselation it lies on. The distance metric used in this example is the Euclidean distance.*

4.2.2   Trustworthiness

Trustworthiness, introduced by Venna et al. [63], is a measure of the degree to which local structure is preserved in the output space relative to the input space after dimensionality reduction is performed. Trustworthiness has been used to compare multiple dimensionality reduction methods for the purposes of visualization of high dimensional data sets such as gene expression data [32] and gene interaction graphs [64].

Computation of trustworthiness is done by comparing how the rank order for each point's k nearest neighbors in the output space match that of the input space. Formally, for a chosen number k of neighbors it is given by the following

$$T(k) = 1 - \frac{2}{nk(2n - 3k - 1)} \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i^k} (r(i,j) - k),$$

where n is the total number of data points under consideration, $\mathcal{N}_i^k$ is the set of k nearest neighbors in the output space of data point i, and $r(i,j)$ is a function giving the rank of node j in the nearest neighbor ordering of node i. The possible values for trustworthiness range from 0 to 1, indicating complete loss of local structure and perfect preservation of local structure, respectively.

4.3   Network Structure and Parameters

To parameterize the embedding $f(\hat{\mathbf{x}})$ a feed-forward neural network with layers of hidden units with dimensions $d - 500 - 500 - 2000 - 2$ were used, where d is the input dimen-

sionality and rectifier linear unit (ReLU) activations are applied to the hidden unit outputs to induce nonlinearity to the network. The hidden unit dimensions were chosen to match those of the final combined network used in [40], although we have substituted the sigmoid activations with ReLU activations. Furthermore, we do not perform stacked restricted Boltzmann machine pretraining on the hidden layer weigths as in [40]. However we do benchmark the performance of our network against the original parametric t-SNE implementation. Since we are not considering RBM pretraining and are instead relying on the property of ReLUs reducing the vanishing gradient problem [23] and better initialization [22]. Running the original implementation shows little benefit in performing this costly initialization. We could instead reuse the trained encoder weights of the VAE as a starting point for the optimization of our embedding network.

For optimization of the neural network [40] use nonlinear conjugate gradient descent, whereas Adam [33] is used as the optimizer in the results presented here. Updates with Adam are considerably faster to compute than with conjugate gradient descent. Throughout all experiments standard parameters for Adam were chosen, i.e. a learning rate of 0.001 and the exponential decay rates of the 1st and 2nd moments were set to 0.9 and 0.999 respectively.

[Note: Should mention recent work on Adam, i.e. the updated version by [50]?]

We apply batch normalization [30] to the hidden layers of the VAE networks, while omitting batch normalization from the embedding network. This is to validate the training procedure on as simple an embedding network as possible. We note that applying batch normalization to the embedding network improves the results marginally.

To further emphasize the universality of our approach we restrict ourselves to use a simple VAE structure, without complicating the model architecture with more recently proposed advances, such as normalizing flows [51] ([Note: could show that better models improve performance in the supplement]). Throughout all experiments the VAE architecture is fixed to $d - 256 - 128 - 32 - \mu, \log \sigma^2 - 32 - 128 - 256 - d$, where μ and σ parameterize a normal distribution acting as the posterior $q_\phi(\mathbf{z}|\mathbf{x}^{(i)})$ and the final layer output was used to parameterize a Bernoulli distribution for MNIST as well as Fashion-MNIST, and a normal distribution with a fixed standard deviation of 0.1 for cytometry data. In all experiments, each VAE was trained with a batch size of 1000 for 10000 iterations.

For MNIST and Fashion-MNIST perplexities were set to 30, which is in line with the value used by [40], whereas for the cytometry data sets a perplexity of 10 was chosen. [Note: Perplexity should really be chosen relative to the batch size. For example training VPTSNE on MNIST with batch size 200 and perplexity 10 gives far better results than with perplexity 30. However, with PTSNE the results are actually worse as the training fails to converge properly, even with small learning rates. An example embedding of MNIST with low perplexity and small batch size has been included in the appendix (figure **??**), which also achieves better trustworthiness (0.935) and 1-NN (0.812) than the 600 batch size, 30

perplexity runs. Analysis of why PTSNE fails to converge and VPTSNE does not should be included, since this is a clear advantage of the method we're demonstrating.]

4.4   Learning

In this section we compare the effect of training on VAE reconstructions to training on the original MNIST data set. To quantitatively evaluate the quality of the embeddings produced we employed the trustworthiness metric [63] on the MNIST test set. Additionally, we compared the 1-NN classification errors by fitting the classifiers on the produced embeddings of the training set and finding the mean accuracies of the classifiers on the test set.

In figure 4.2 we have plotted the 1-NN scores and trustworthiness of the embeddings obtained after each iteration of training for two different batch sizes. The runs for each batch size were repeated 20 times, plotting the means and 95% confidence intervals. Training with VAE reconstructions shows a clear improvement both qualitatively and quantitatively over training on the original data when small training batches are used to approximate the t-SNE loss gradient. Higher trustworthiness, as well as 1-NN scores are obtained consistently and convergence is reached with far fewer iterations. Moreover, the results are in favor of training on reconstructions by exhibiting more stable results during training.

Qualitatively the embeddings trained on the original data remain noisier than the embeddings trained on the reconstructed data points. This can be seen in figure 4.3, where the separation of true clusters in the embedding is far less evident, with several classes overlapping and a large number of outlier points for each class. We further compare the qualitative differences of the embeddings produced with the Fashion-MNIST dataset. In figure 4.4 similar deficiencies in the embedding can be noted as in the embedding comparisons for the MNIST data set. In particular, the visible clusters of classes are less distinct, as well as the global layout of the classes is considerably worse, e.g. the cluster of footwear related images has been pulled closer to the cluster of upper body garments and the images of bags have been split into two seemingly unrelated clusters.

??? Plot like Figure 4.2 for VPTSNE vs PTSNE with batch size on x to show that VPTSNE does not suffer that much from small batch sizes? This would imply that it can be trained with smaller batch sizes, which means better asymptotic performance. ???

4.5   Comparisons

We now compare the proposed method to other parametric dimensionality reduction methods, namely PCA and VAE. In addition we also consider two non-parametric dimensionality reduction methods, t-SNE and the recently proposed method UMAP [45], which had been demonstrated to be able to produce embeddings comparable to t-SNE while being more efficient to optimize in practice.
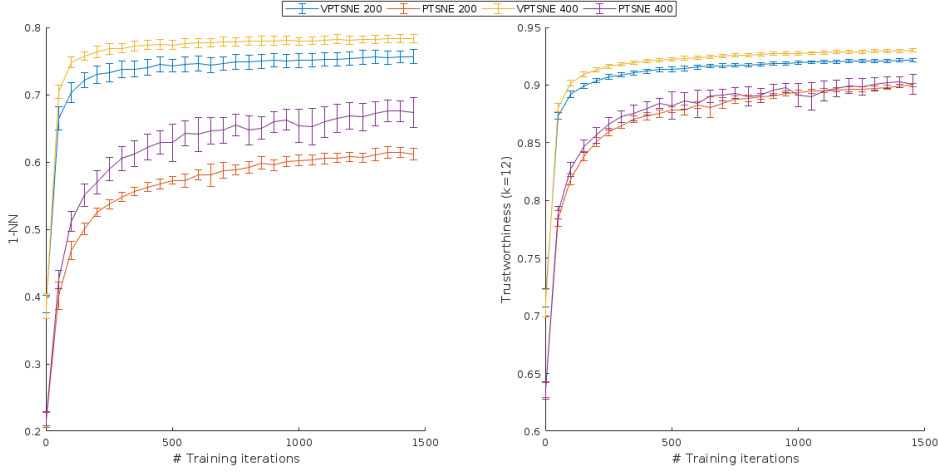
Figure 4.2. *Plots of 1-NN and trustworthiness scores obtained after a given number of iterations for different batch sizes. The means and 95% confidence intervals have been plotted from 20 repeated runs for each parameter setting.*
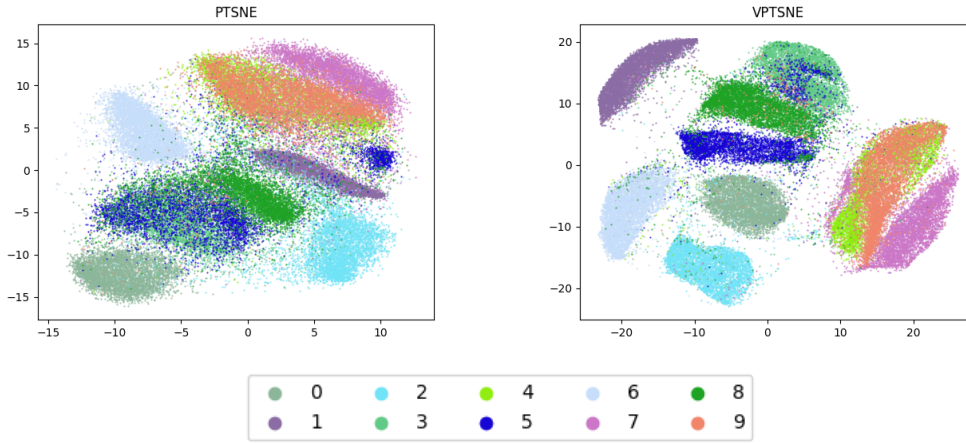
Figure 4.3. *Embeddings of the MNIST data set trained with batch size 400.*

For the comparisons we will be using the Cytometry A data set. As evaluation criteria, similarly to the previous section, both trustworthiness and 1-NN classification accuracy are used. In addition to comparing the quality of the embeddings, the runtime performance of the different methods are studied. Our goal is to demonstrate that our method produces competitive embeddings as well as has superior scalability.

4.5.1   Embedding Quality

The quantitative results for all methods under comparison are available in table 4.1 and the corresponding scatter plots of the 2-D embeddings are presented in figure 4.5. Although VAE produces quantitatively better results for the chosen metrics, the spatial layout of clusters in the resulting embedding have less distinct separation and the global structure of the embedding is constrained by the chosen prior $p(\mathbf{z})$. From a data visualization
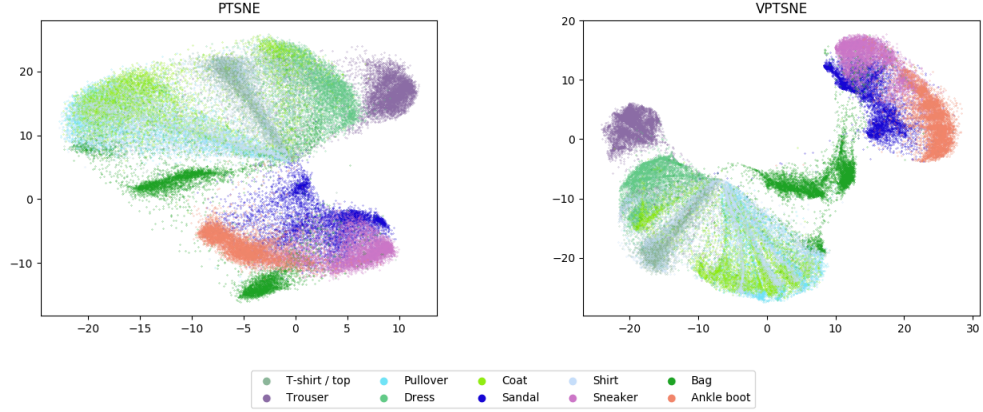
Figure 4.4. *Embeddings of the Fashion-MNIST data set trained with batch size 400. [Note: could include UMAP embedding of this data set here for comparison. It is remarkably similar and this would reinforce the claim that small batch size training is actually very effective.]*

standpoint these factors make the latent space embeddings qualitatively worse than what the corresponding quantitative metrics would suggest. Unsurprisingly, PCA performs the worst on both quantitative and qualitative results. Our method however is able to reach results on par with those of the chosen non-parametric methods.

[Note: The effect of choosing a suitable perplexity w.r.t. batch size mentioned in a previous note is highlighted in 4.5. With large perplexity the clusters in the t-SNE embedding begin to separate more distinctly like with VPTSNE/UMAP, but here the same perplexity is used for both VPTSNE and t-SNE optimization. As perplexity is increased t-SNE optimization becomes considerably slower (for example going from perplexity 10 to 100 increased the total computation time twofold).]

[Note: We can also observe that t-SNE (at least with the given perplexity) preserves global structure much more poorly. This can be observed by comparing with the PCA plot, which has three clearly distinct, spatially separated clusters of clusters that are not present in the t-SNE plot. Could consider using some metric to quantify this?]

4.5.2 Scalability

A major practical challenge with nonlinear dimensionality reduction methods are their scalability to large data sets. Typically, to capture the spatial structure of data a distance metric is employed to compute the pairwise distances for the entire data set, after which a projection of the points to a lower dimensional space, equipped with a corresponding metric, is sought that retains the computed distances as accurately as possible. As the number of data points increases, the inherent quadratic complexity of methods relying on such pairwise distance computations quickly renders such methods intractable.

In practice, various approximations and data structures are employed to speed up di-

Table 4.1. *Comparison between different dimensionality reduction methods.*

Algorithm	Trustworthiness ($k = 12$)	1-NN	time (ms)
VPTSNE	0.9753	0.9250	60766
PTSNE	0.9628	0.8890	8729
UMAP	0.9688	0.9297	74716
VAE	0.9762	0.9247	52037
t-SNE	0.9879	0.9536	1192812
PCA	0.8557	0.4886	96

mensionality reduction methods. With t-SNE a widely used performance enhancement, independently investigated by [68] and [41], is to use the Barnes-Hut algorithm [5] to approximate the Q_{ij} matrix by averaging the influence of distant points belonging to the same node in the underlying space partitioning data structure. This approximation nevertheless fails to provide significant speedup in the general case depending on the chosen tradeoff between accuracy and speed, as well as the distribution of the points being optimized at each step. Furthermore, its use is limited to obtaining 2 or 3 dimensional embeddings due to its use of the quadtree and octree data structures for each respective dimensionality. On the other hand, optimization of the parametric embedding can be performed with stochastic gradient descent. As we have shown, even with small batch sizes we are able to produce embeddings of competitive quality. The Barnes-Hut approximation can also be used to further speed up the computation of the loss and its gradient, but for small batch sizes the computational overhead of constructing the required data structure can outweigh the cost of directly computing all pairwise influences within the batch. Direct computation has the additional benefit that it is not limited to output dimensionalities for which a space partitioning data structure can be efficiently built.

As previously mentioned, one of the strengths of UMAP compared to t-SNE is the efficiency of its optimization. The key to its efficient optimization is that its gradient can be approximated stochastically via probabilistic edge sampling following the procedure introduced in [60]. As both VPTSNE and UMAP take advantage of stochastic gradient descent, the scalability of their optimization is equal up to an implementation dependent constant factor. VPTSNE however maintains the advantage over UMAP that natural out-of-sample extension is made possible through the learned embedding function.

4.6   Robustness to Sparse Data

As discussed in section [todo and ref. the correct section in methods] the probabilistic model is advantageous when only few data points are available. To show this effect in practice we trained mappings with and without the use of our method on randomly chosen subsets of the MNIST training data set, effectively simulating sparsity of available training data. Evaluation of the mappings was carried out on the full MNIST test set as in previous experiments. A line plot of the obtained scores corresponding to the chosen size of the subsets can be found in figure 4.6. Significant differences in the obtained scores are noted
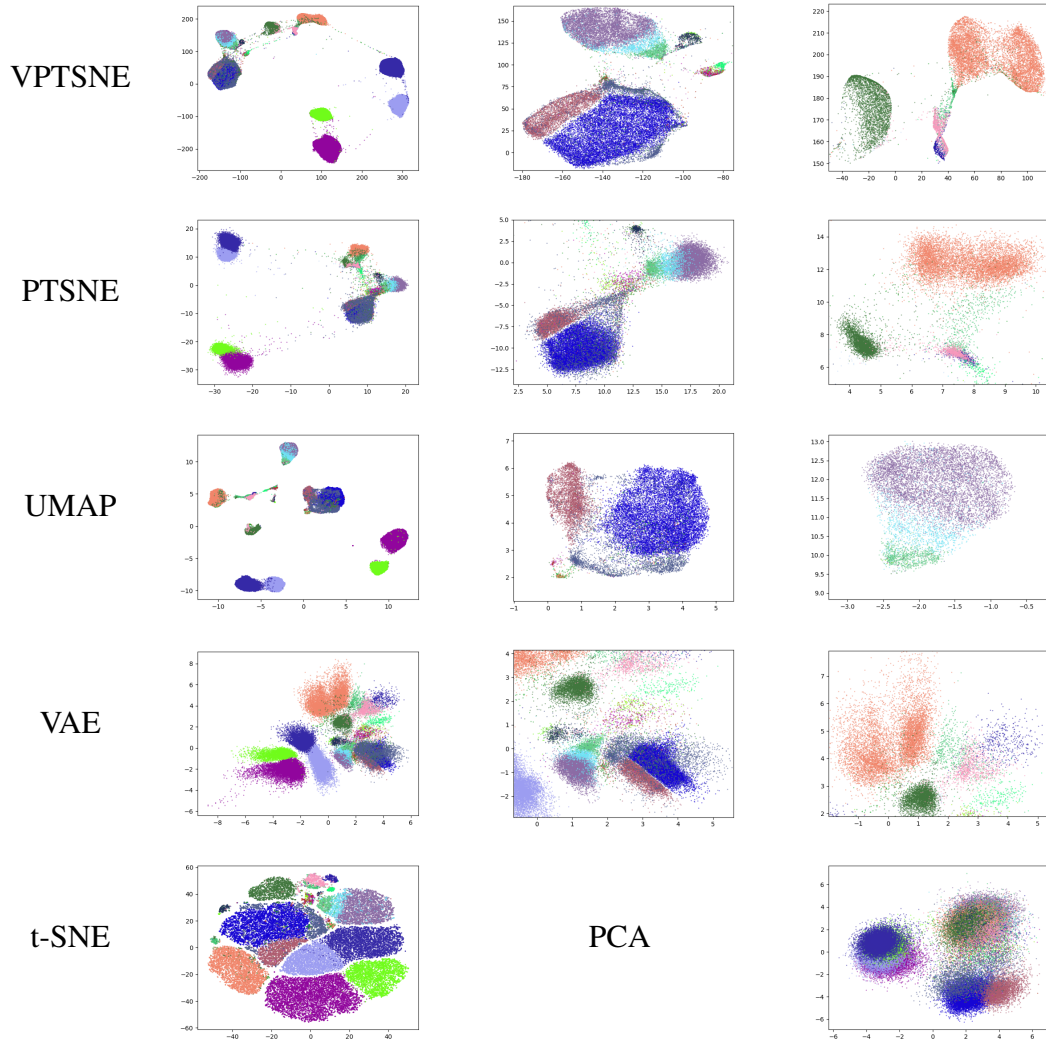
Figure 4.5. *Cytometry A data set embedded with the methods being compared in section 4.5.*

in favor of our method. From these results we can observe that both the trustworthiness and 1-NN scores remain higher even with extremely small data sets, indicating that the use of our method is able to provide better generalization even when few training data points are available.

4.7 Robustness to Noisy Data

As real world data is generally not perfectly clean, an important property for any machine learning method is its capability to handle data containing artifacts. Considering VAEs are inherently robust to corrupted data due to the regularization provided by the stochastic latent code ([should be discussed in a previous section more thoroughly, connection with VAE and RPCA [10, 11] shown in [14]]) it is reasonable to assume that a training procedure for a low dimensional mapping taking advantage of this property would perform better on corrupted data than one that is subjected only to the raw data.

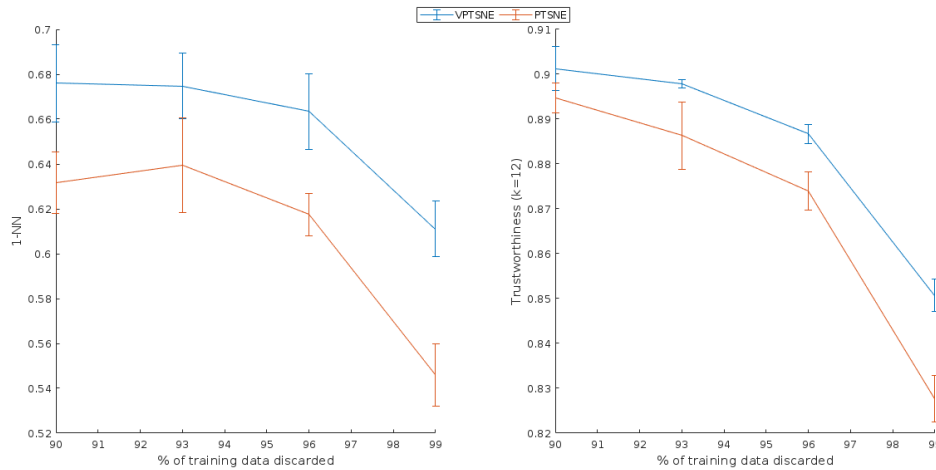***Figure 4.6****. 1-NN and trustworthiness scores obtained for PTSNE and VPTSNE trained on small subsets of the MNIST data set. Error bars indicate the 95% confidence intervals of the mean for each sparsity level, obtained from 20 repeated runs.*

Here we aim to back up this assumption by running experiments on artificially corrupted MNIST data sets. As the corruption process in our experiments we consider *masking noise* as in [65], where a predetermined fraction of randomly chosen elements of a data point are set to 0. We test robustness by applying masking noise to 10%, 20%, 30% and 40% fractions of the training and test sets. The results of our experiments (cf. figure 4.7) show a clear advantage to using our method when dealing with noisy data as both evaluation metrics can be observed to decrease considerably for PTSNE as the training data is progressively corrupted, whereas VPTSNE maintains its performance throughout the experiments.

[Note: Results look good, but could they be further improved by denoising variational autoencoders [29]?]

[Note: In scRNA-seq dropout events can be viewed as masking noise. The results here would indicate applying the method to scRNA-seq data could produce good results.]

[Note: Should other corruption processes be considered here or is masking noise sufficient?]

[Note: Could add scatter plot of the best scoring embeddings PTSNE vs VPTSNE to the supplement for e.g. the 30% corrupted runs. The difference is quite striking.]

Again, I would want more points on the graphs. Noise level on x axis and score on y.

4.8   Obtaining Reconstructions from Hidden Layers

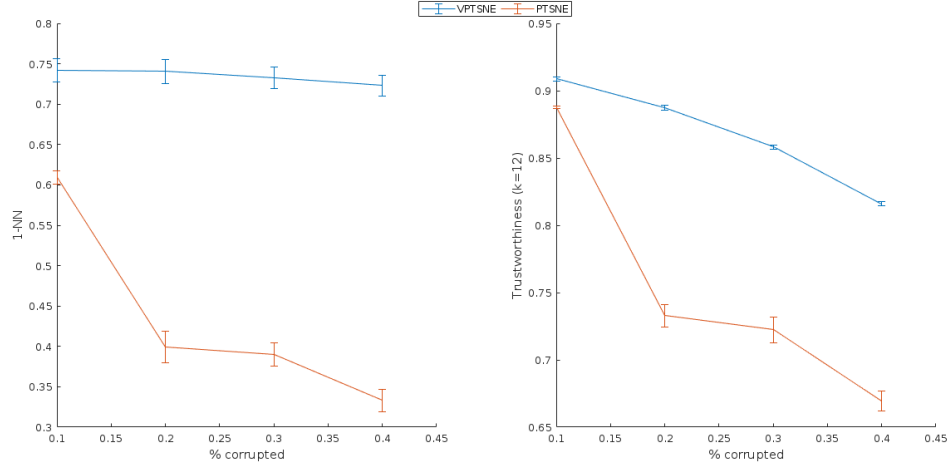[Note: Much of this will probably be moved to the methods section.]

Figure 4.7. *1-NN and trustworthiness scores obtained for PTSNE and VPTSNE trained on MNIST data with different levels of masking noise applied. Error bars indicate the 95% confidence intervals of the mean for each corruption level, obtained from 20 repeated runs.*

Instead of training the embedding network on the final output of the observation model it is possible to instead use the outputs of a chosen hidden layer. If the chosen hidden layer has dimensions considerably smaller than that of the original, the benefits of this approach are twofold:

- We are performing a step of nonlinear dimensionality reduction that is not dependent on local distance metrics, i.e. a preprocessing step that is less susceptible to the *curse of dimensionality*.

- Given that the number of dimensions can be chosen to be considerably lower than in the original input space, computing the t-SNE loss also becomes proportionally cheaper.

To show the efficacy of this modification to the learning procedure we train an embedding using the latent code directly. As we choose the dimensionality of the latent code to be more than two orders of magnitude smaller than that of the original data's it is possible for us to use batches of far greater size to train the embedding network, without incurring additional computation cost over training with smaller batches on data of the original dimensionality. The latent dimensionality of the VAE was chosen to be 7 and the batch size for training the embedding was set to 5000, as in the original parametric t-SNE paper by [40]. We have additionally included the embedding of data points obtained with the original implementation of parametric t-SNE for comparison in figure 4.8. Qualitatively and quantitatively we achieve an overall better embedding with less computation using our method.

When considering precomputed P_{ij} matrices for each batch the gain in computational advantage does however diminish significantly, although the following caveats apply to the precomputation: Storing the precomputed matrices will require considerable space

for large batch sizes and data sets. A way to circumvent the need for large storage space is to employ a KNN approximation when computing P_{ij}, i.e. only considering a small number of neighbors when computing the conditional probabilities for each pair of points, which will result in a sparse matrix. This however increases the error in the computed gradient and relies on an efficient KNN algorithm in practice. Additionally, fixing the batches in advance hinders the performance of SGD as the gradient estimates become biased. Moreover, relying on precomputed P_{ij} matrices for fixed batches prevents the use of infinite sample generation from the VAE.
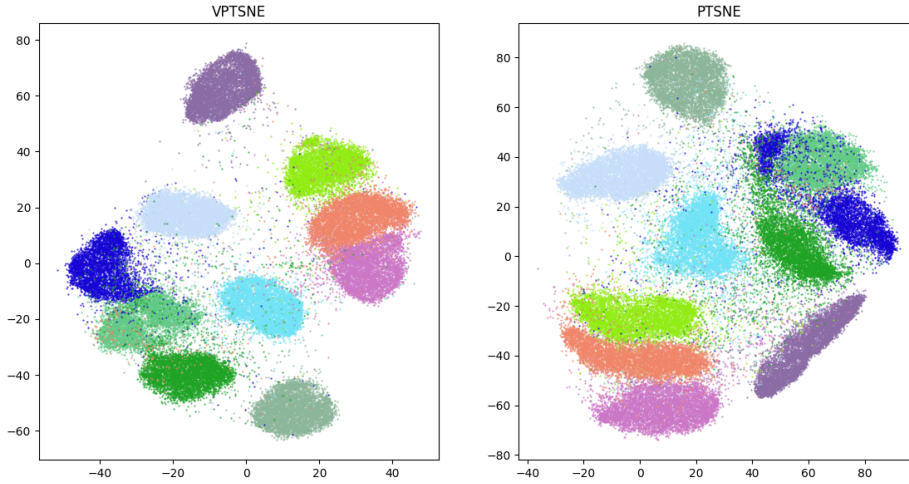

Figure 4.8. *Embedding produced by training on a 7 dimensional latent representation of the MNIST data set contrasted with the original implementation of [40] run on the original data. VPTSNE trustworthiness: 0.927, 1-NN: 0.910. PTSNE trustworthiness: 0.926, 1-NN 0.887.*

[Note: The choice of 7 as the latent dimensionality right now seems unjustified. It was simply chosen to be the largest number for which "more than two orders of magnitude smaller" holds true. Should work that into the text.]

4.9   Inference with the Generative Model

From the underlying generative model we can approximately compute the marginal $p(\mathbf{x})$ for each data point $\mathbf{x}$. An approximation can be retrieved by importance sampling [52]:

$$p(\mathbf{x}) \approx \frac{1}{N} \sum_{i=1}^{N} \frac{p(\mathbf{x}|\mathbf{z}^{(i)})p(\mathbf{z}^{(i)})}{q_\theta(\mathbf{z}^{(i)}|\mathbf{x})},$$

where $\mathbf{z}^{(i)} \sim q_\theta(\mathbf{z}|\mathbf{x})$.

Using the obtained marginals as feedback we can inform the user of potentially anomalous points in the data set. This feedback can then further be used in downstream analysis. From

a data visualization perspective points that do not fit the model well correspond to points that are likely to be mapped poorly, thus removing such outliers from the final mapping serves to highlight the natural clusters in the data more clearly.

[Note: discuss the more accurate but computationally costly marginal estimation method: annealed importance sampling [47, 66]?]

[Note: could discuss, experiment with modification to the algorithm: if p(x) (or even more simply just the reconstruction error) is high (above some threshold), pass the original data point and not the reconstruction to the mapping. This proved to work for the small subset experiment when limiting the latent dimension to ≤ 3. Reason for this being that the VAE trained on only 600 samples poorly reconstructed the test set (basically just corrupting many of them).]

[Note: contents above should be moved to the methods section.]

As an example we use the Cytometry B data set to train our combined model with two patients' data held out. The protein expression profile of Patient A's cells closely match that of other patients' in the training set, whereas Patient B's sample is from advanced cancer making it an outlier in relation to the data available during training.

To distinguish between outlier and inlier points in the mapping we obtain we set a hard threshold of $\log p(\mathbf{x}) < -150$ to separate between the two. In figure 4.9 embeddings with both patient specific labeling and labeling obtained by thresholding are shown side by side. The high tumor purity sample of Patient B has almost completely been mapped to a small region corresponding to protein expression levels of cancer cells, whereas Patient A's mapping contains several distinct clusters of various cell types. Further, the labels obtained through thresholding indicate that the majority of cells in Patient B's sample are indeed considered as outliers by the model, in addition to a handful of visually poorly mapped points.
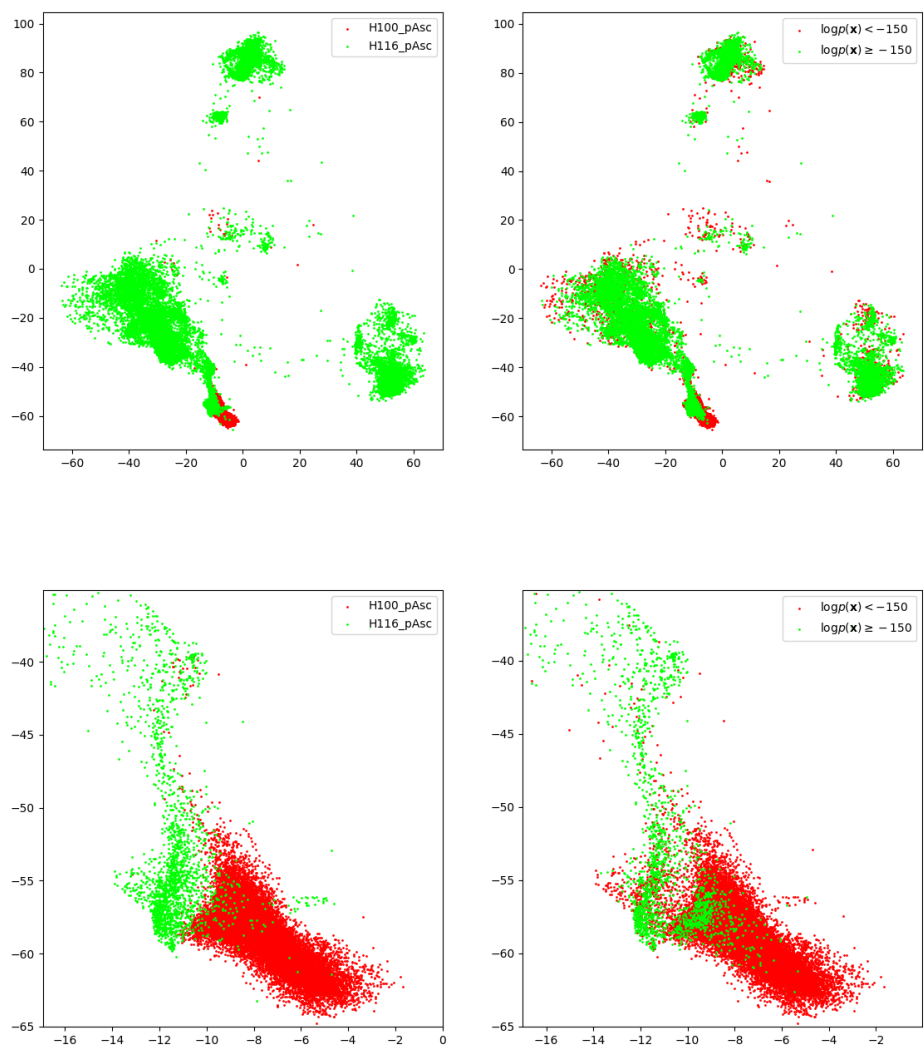
Figure 4.9*. Detecting outliers in cytometry data. [Rename patients to match text.]*

5. CONCLUSIONS

[30, 51, 21, 40, 42, 52, 35, 19, 3, 39, 4, 49, 61, 34, 63, 45, 38, 23, 22, **?, ?**, 59, 24, 55, 68, 41, 62, 26]

REFERENCES

[1]   M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, Tensorflow: a system for large-scale machine learning., in: OSDI, 2016, Vol. 16, pp. 265–283.

[2]   C.C. Aggarwal, A. Hinneburg, D.A. Keim, On the surprising behavior of distance metrics in high dimensional space, in: International conference on database theory, Springer, 2001, pp. 420–434.

[3]   N. Aghaeepour, G. Finak, H. Hoos, T.R. Mosmann, R. Brinkman, R. Gottardo, R.H. Scheuermann, Critical assessment of automated flow cytometry data analysis techniques, Nature Methods, Vol. 10, Iss. 3, Feb. 2013, pp. 228–238.

[4]   Amir El-ad David, Davis Kara L, Tadmor Michelle D, Simonds Erin F, Levine Jacob H, Bendall Sean C, Shenfeld Daniel K, Krishnaswamy Smita, Nolan Garry P, Pe'er Dana, viSNE enables visualization of high dimensional single-cell data and reveals phenotypic heterogeneity of leukemia, Nature biotechnology, Vol. 31, Iss. 6, may, 2013, p. 545–552.

[5]   Barnes Josh, Hut Piet, A hierarchical O(N log N) force-calculation algorithm, Nature, Vol. 324, dec, 1986, p. 446.

[6]   R. Bellman, R. Bellman, R. Corporation, Dynamic Programming, Princeton University Press, Rand Corporation research study, 1957.

[7]   D.M. Blei, A. Kucukelbir, J.D. McAuliffe, Variational inference: A review for statisticians, Journal of the American Statistical Association, Vol. 112, Iss. 518, 2017, pp. 859–877.

[8]   C.J.C. Burges, Dimension reduction: A guided tour, Foundations and Trends® in Machine Learning, Vol. 2, Iss. 4, 2010, pp. 275–365.

[9]   C.P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, A. Lerchner, Understanding disentangling in β-VAE, ArXiv e-prints, Apr. 2018.

[10]   E.J. Candès, X. Li, Y. Ma, J. Wright, Robust principal component analysis?, Journal of the ACM (JACM), Vol. 58, Iss. 3, 2011, p. 11.

[11]   V. Chandrasekaran, S. Sanghavi, P.A. Parrilo, A.S. Willsky, Rank-sparsity incoherence for matrix decomposition, SIAM Journal on Optimization, Vol. 21, Iss. 2, 2011, pp. 572–596.

[12]   T.Q. Chen, X. Li, R. Grosse, D. Duvenaud, Isolating sources of disentanglement in variational autoencoders, arXiv preprint arXiv:1802.04942, 2018.

[13]   J. Cook, I. Sutskever, A. Mnih, G. Hinton, Visualizing similarity data with a mixture of maps, in: Artificial Intelligence and Statistics, 2007, pp. 67–74.

[14]   B. Dai, Y. Wang, J. Aston, G. Hua, D. Wipf, Connections with robust pca and the role of emergent sparsity in variational autoencoder models, Journal of Machine Learning Research, Vol. 19, Iss. 41, 2018, pp. 1–42.

[15]   J. Deng, W. Dong, R. Socher, L.J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, Ieee, 2009, pp. 248–255.

[16]   J.V. Dillon, I. Langmore, D. Tran, E. Brevdo, S. Vasudevan, D. Moore, B. Patton, A. Alemi, M. Hoffman, R.A. Saurous, Tensorflow distributions, arXiv preprint arXiv:1711.10604, 2017.

[17]   W. Dong, C. Moses, K. Li, Efficient k-nearest neighbor graph construction for generic similarity measures, in: Proceedings of the 20th international conference on World wide web, ACM, 2011, pp. 577–586.

[18]   C. Fefferman, S. Mitter, H. Narayanan, Testing the manifold hypothesis, Journal of the American Mathematical Society, Vol. 29, Iss. 4, 2016, pp. 983–1049.

[19]   B.J. Frey, D. Dueck, Clustering by passing messages between data points, Science, Vol. 315, Iss. 5814, 2007, pp. 972–976.

[20]   K.P. F.R.S., LIII. On lines and planes of closest fit to systems of points in space, The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, Vol. 2, Iss. 11, 1901, pp. 559–572.

[21]   A. Gisbrecht, A. Schulz, B. Hammer, Parametric nonlinear dimensionality reduction using kernel t-SNE, Neurocomputing, Vol. 147, Advances in Self-Organizing Maps Subtitle of the special issue: Selected Papers from the Workshop on Self-Organizing Maps 2012 (WSOM 2012), 2015, pp. 71–82.

[22]   X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Teh, Y.W., Titterington, M. (eds.), Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, Chia Laguna Resort, Sardinia, Italy, 13–15 May, 2010, Proceedings of Machine Learning Research 9, PMLR, pp. 249–256.

[23]   X. Glorot, A. Bordes, Y. Bengio, Deep sparse rectifier neural networks, in: JMLR W&CP: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2011), Apr. 2011.

[24]   I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative Adversarial Nets, in: Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q. (eds.), Advances in Neural Information Processing Systems 27, Curran Associates, Inc., 2014, pp. 2672–2680. Available: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

[25]   I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, A. Lerchner, beta-vae: Learning basic visual concepts with a constrained variational framework, 2016.

[26]   G.E. Hinton, S.T. Roweis, Stochastic neighbor embedding, in: Advances in neural information processing systems, 2003, pp. 857–864.

[27]   K. Hornik, Approximation capabilities of multilayer feedforward networks, Neural networks, Vol. 4, Iss. 2, 1991, pp. 251–257.

[28]   H. Hotelling, Relations between two sets of variates, Biometrika, Vol. 28, Iss. 3/4, 1936, pp. 321–377.

[29]   D.J. Im, S. Ahn, R. Memisevic, Y. Bengio, Denoising criterion for variational auto-encoding framework, in: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA., 2017, pp. 2059–2065.

[30]   S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, CoRR, Vol. abs/1502.03167, 2015.

[31]   W.B. Johnson, J. Lindenstrauss, Extensions of lipschitz mappings into a hilbert space, Contemporary mathematics, Vol. 26, Iss. 189-206, 1984, p. 1.

[32]   S. Kaski, J. Nikkilä, M. Oja, J. Venna, P. Törönen, E. Castrén, Trustworthiness and metrics in visualizing similarity of gene expression, BMC Bioinformatics, Vol. 4, Iss. 1, Oct, 2003, p. 48.

[33]   D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, CoRR, Vol. abs/1412.6980, 2014.

[34]   D.P. Kingma, T. Salimans, M. Welling, Improving variational inference with inverse autoregressive flow, CoRR, Vol. abs/1606.04934, 2016.

[35]   D.P. Kingma, M. Welling, Auto-encoding variational bayes., CoRR, Vol. abs/1312.6114, 2013.

[36]   A. Kucukelbir, D. Tran, R. Ranganath, A. Gelman, D.M. Blei, Automatic differentiation variational inference, The Journal of Machine Learning Research, Vol. 18, Iss. 1, 2017, pp. 430–474.

[37]   Y. LeCun, Y. Bengio, G. Hinton, Deep learning, nature, Vol. 521, Iss. 7553, 2015, p. 436.

[38]   Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Vol. 86, Dec. 1998, pp. 2278 – 2324.

[39]   Levine Jacob H., Simonds Erin F., Bendall Sean C., Davis Kara L., Amir El-ad D., Tadmor Michelle D., Litvin Oren, Fienberg Harris G., Jager Astraea, Zunder Eli R., Finck Rachel, Gedman Amanda L., Radtke Ina, Downing James R., Pe'er Dana, Nolan Garry P., Data-Driven Phenotypic Dissection of AML Reveals Progenitor-like Cells that Correlate with Prognosis, Cell, Vol. 162, Iss. 1, doi: 10.1016/j.cell.2015.05.047, feb, 2018, pp. 184–197.

[40]   L. Maaten, Learning a parametric embedding by preserving local structure, in: Dyk, D. van, Welling, M. (eds.), Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics, Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA, 16–18 Apr, 2009, Proceedings of Machine Learning Research 5, PMLR, pp. 384–391.

[41]   L. van der Maaten, Accelerating t-SNE using Tree-Based Algorithms, Journal of Machine Learning Research, Vol. 15, 2014, pp. 3221–3245.

[42]   L. van der Maaten, G. Hinton, Visualizing data using t-SNE, Journal of Machine Learning Research, Vol. 9, 2008, pp. 2579–2605.

[43]   L. Van der Maaten, G. Hinton, Visualizing non-metric similarities in multiple maps, Machine learning, Vol. 87, Iss. 1, 2012, pp. 33–55.

[44]   L. van der Maaten, E. Postma, H. van den Herik, Dimensionality reduction: A comparative review.

[45]   L. McInnes, J. Healy, UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction, ArXiv e-prints, Feb. 2018.

[46]   V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski *et al.*, Human-level control through deep reinforcement learning, Nature, Vol. 518, Iss. 7540, 2015, p. 529.

[47]   R.M. Neal, Annealed importance sampling, Statistics and computing, Vol. 11, Iss. 2, 2001, pp. 125–139.

[48]   J. Nickolls, I. Buck, M. Garland, K. Skadron, Scalable parallel programming with cuda, Queue, Vol. 6, Iss. 2, Mar. 2008, pp. 40–53.

[49]   A. Nima, N. Radina, H. Hoos Holger, R. Brinkman Ryan, Rapid Cell Population Identification in Flow Cytometry Data, Cytometry. Part A : the journal of the International Society for Analytical Cytology, Vol. 79, Iss. 1, jan, 2011, p. 6–13.

[50] S.J. Reddi, S. Kale, S. Kumar, On the convergence of adam and beyond, in: International Conference on Learning Representations, 2018.

[51] D. Rezende, S. Mohamed, Variational inference with normalizing flows, in: Bach, F., Blei, D. (eds.), Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 07–09 Jul, 2015, Proceedings of Machine Learning Research 37, PMLR, pp. 1530–1538.

[52] D.J. Rezende, S. Mohamed, D. Wierstra, Stochastic backpropagation and approximate inference in deep generative models, in: Xing, E.P., Jebara, T. (eds.), Proceedings of the 31st International Conference on Machine Learning, Bejing, China, 22–24 Jun, 2014, Proceedings of Machine Learning Research 32, PMLR, pp. 1278–1286.

[53] V. Rokhlin, Rapid solution of integral equations of classical potential theory, Journal of computational physics, Vol. 60, Iss. 2, 1985, pp. 187–207.

[54] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representations by back-propagating errors, nature, Vol. 323, Iss. 6088, 1986, p. 533.

[55] S. Santurkar, D. Tsipras, A. Ilyas, A. Madry, How Does Batch Normalization Help Optimization? (No, It Is Not About Internal Covariate Shift), ArXiv e-prints, May 2018.

[56] S. Shalev-Shwartz, S. Ben-David, Understanding machine learning: From theory to algorithms, Cambridge university press, 2014.

[57] D. Silver, A. Huang, C.J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, Mastering the game of go with deep neural networks and tree search, nature, Vol. 529, Iss. 7587, 2016, p. 484.

[58] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, Mastering the game of go without human knowledge, nature, Vol. 550, Iss. 7676, 2017, p. 354.

[59] M.H. Spitzer, G.P. Nolan, Mass Cytometry: Single Cells, Many Features, Cell, Vol. 165, Iss. 4, 2016, pp. 780–791.

[60] J. Tang, J. Liu, M. Zhang, Q. Mei, Visualizing large-scale and high-dimensional data, in: Proceedings of the 25th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee, 2016, pp. 287–297.

[61] J.M. Tomczak, M. Welling, Improving variational auto-encoders using householder flow, CoRR, Vol. abs/1611.09630, 2016.

[62] L. Van Der Maaten, K. Weinberger, Stochastic triplet embedding, in: Machine Learning for Signal Processing (MLSP), 2012 IEEE International Workshop on, IEEE, 2012, pp. 1–6.

[63] J. Venna, S. Kaski, Neighborhood preservation in nonlinear projection methods: An experimental study, in: Dorffner, G., Bischof, H., Hornik, K. (eds.), Artificial Neural Networks — ICANN 2001, Berlin, Heidelberg, 2001, Springer Berlin Heidelberg, pp. 485–491.

[64] J. Venna, S. Kaski, Visualizing gene interaction graphs with local multidimensional scaling, in: ESANN, 2006.

[65] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.A. Manzagol, Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion, Journal of machine learning research, Vol. 11, Iss. Dec, 2010, pp. 3371–3408.

[66] Y. Wu, Y. Burda, R. Salakhutdinov, R.B. Grosse, On the quantitative analysis of decoder-based generative models, CoRR, Vol. abs/1611.04273, 2016.

[67] H. Xiao, K. Rasul, R. Vollgraf, Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[68] Z. Yang, J. Peltonen, S. Kaski, Scalable optimization of neighbor embedding for visualization, in: International Conference on Machine Learning, 2013, pp. 127–135.

[69] S. Zhao, J. Song, S. Ermon, Infovae: Information maximizing variational autoencoders, arXiv preprint arXiv:1706.02262, 2017.