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, 46 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, 46 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
- High dimensional data visualization, explorative data analysis
  - Common in fields such as ...
  - Unsupervised learning
- Previous work
  - Describe common dimensionality reduction methods, pros and cons
  - Parametric vs non-parametric methods
- Contribution of this thesis
  - Enhance parametric t-SNE, so that ...
- Layout of this thesis
  - What is covered in each chapter
  - What background is expected from the reader

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 of high dimensional data, several problems arise

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. - algorithms depend on dimensionality of the data being processed and large computational costs are accrued, especially sampling becomes - In literature, problems arising from high dimensionality are collectively referred to as the *curse of dimensionality*. feature selection and feature extraction as solutions. in this work we focus on purely unsupervised feature extraction through

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 ... [11] and ... [17] for the study of ... . PCA however suffers from the fact that it only aims to maximize the explained variance... thus local structure of the data is poorly preserved. Methods that aim to preserve local structure ...

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

A further issue is the volume of data being gathered. Most methods scale poorly to large amounts of data, leaving

In this work we demonstrate a method for improving the training of ptsne.

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 ... . 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 the t-Distributed Stochastic Neighbor Embedding algorithm (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 for this thesis. Additionally, directions for future work are suggested.

2. BACKGROUND AND RELATED WORK

2.1 Variational Autoencoders

2.1.1 Artificial Neural Networks

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

Basic Principles

Learning in Artificial Neural Networks

Modern Advances and Best Practices

2.1.2 Autoencoders

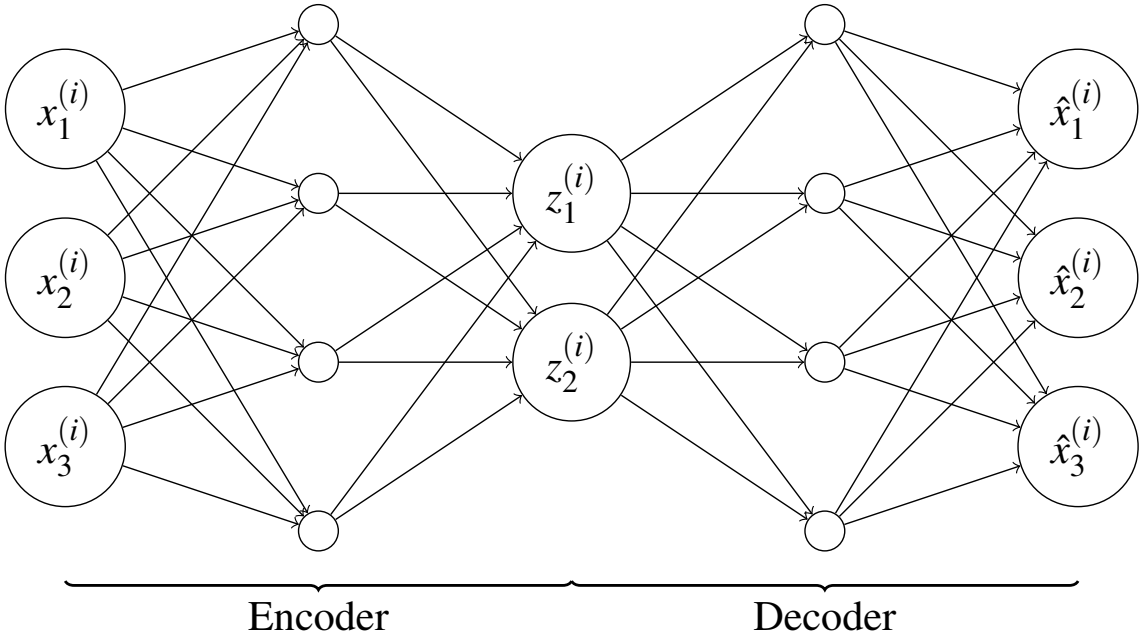

Figure 2.1. *Autoencoder*

2.1.3 Variational Bayesian Methods

Kullback-Leibler Divergence

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 for discrete probability distributions

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

The Evidence Lower Bound

2.1.4 Auto Encoding Variational Bayes

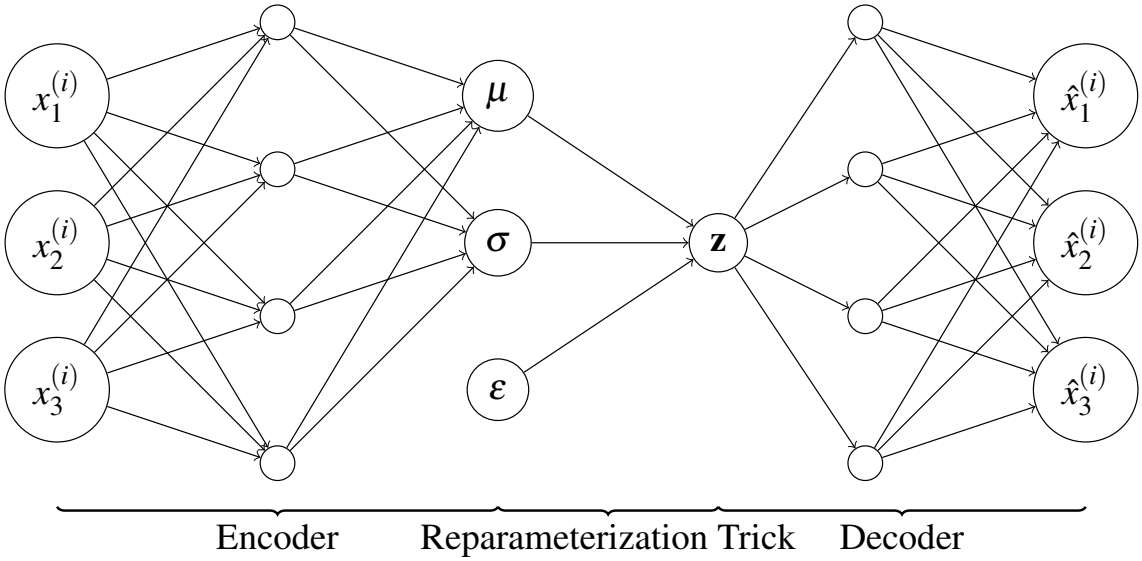

Figure 2.2. *Variational Autoencoder*

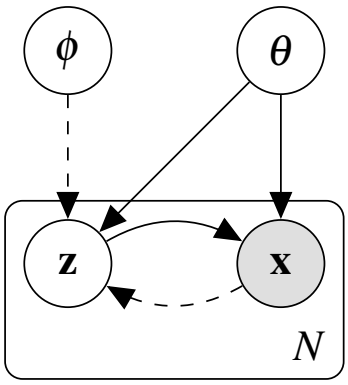

Figure 2.3. *AEVB plate notation*

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:

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

This bound is better known as the *evidence lower bound*, or ELBO for short. 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.

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

2.2.1 SNE

2.2.2 t-SNE

$$2^{H(p(\mathbf{x}))} = 2^{-\sum_{i=1}^{N} p(\mathbf{x}^{(i)}) \log_2 p(\mathbf{x}^{(i)})}$$

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

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

$$C = D_{\mathrm{KL}}(P \parallel Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

$$\frac{\partial C}{\partial \mathbf{y}_i} =$$

2.2.3 Parametric t-SNE

3. METHOD

3.1 Learning a Parametric Embedding Using VAE Sampling

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 [8] 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 [30] kernel operations.

[Note: Describe that it also includes fast parallel implementation of NN-Descent [9]?]

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 [23] 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 an 1-NN classifier.

4.1.2 Fashion-MNIST

Move to supplement? Currently only used in section 4.4. Ref. [43]

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 on mass cytometry and associated analytical challenges the reader is referred to [36].

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 [24], 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

4.2.1 Nearest Neighbor Classifier

4.2.2 Trustworthiness

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 $d \times 28 - 500 - 500 - 2000 - 2$ were used, where d is the input dimensionality 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 [25], 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 [25]. 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 [14] and better initialization [13]. 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 [25] use nonlinear conjugate gradient descent, whereas Adam [20] is used as the optimizer in our results. 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 [32]?]

We apply batch normalization [19] 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 [33] ([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.

The perplexities used for each data set were chosen such that the trained embeddings produced the best results. For MNIST and Fashion-MNIST perplexities were set to 30, which is in line with the value used by [25], whereas for the cytometry data sets a perplexity of 10 was chosen.

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 [40] 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 4.1 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 4.2, 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 4.3 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 4.1 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, VAE and Autoencoders. In addition we also consider two non-parametric dimensionality reduction methods, t-SNE and the recently proposed method UMAP [28], which had been demonstrated to be able to produce embeddings comparable to t-SNE while being more efficient to optimize in practice.

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

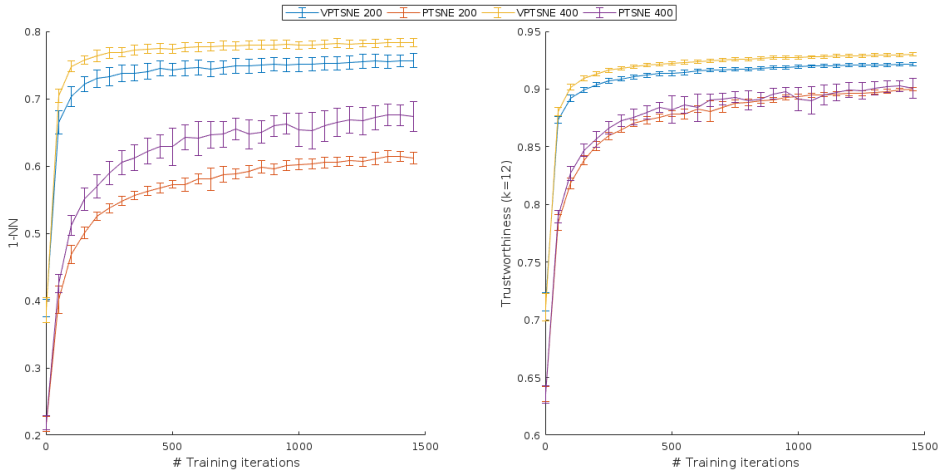

Figure 4.1. *Plots of 1-NN and trustworthiness scores obtained after a given number of iterations for different batch sizes. The means and 95% confidence intervals have been plotted from 20 repeated runs for each parameter setting.*

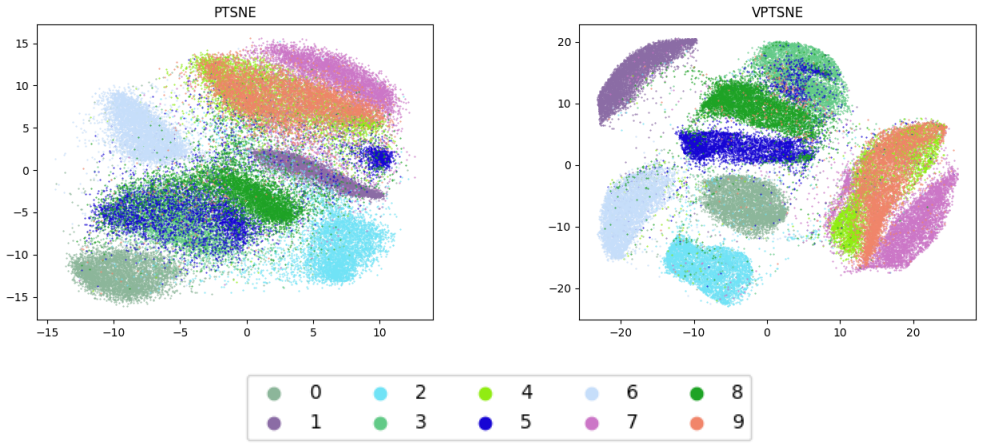

Figure 4.2. *Embeddings of the MNIST data set trained with batch size 400.*

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 4.4. 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 standpoint these factors make the latent space embeddings qualitatively worse than what the corresponding quantitative metrics would suggest. Unsurprisingly, PCA performs the

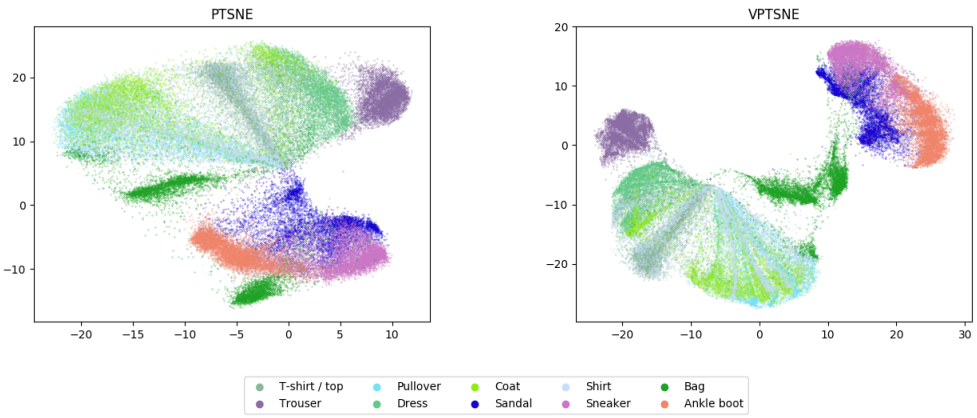

Figure 4.3. *Embeddings of the Fashion-MNIST data set trained with batch size 400. [Note: could include UMAP embedding of this data set here for comparison. It is very similar and this would reinforce the claim that small batch size training is actually very effective.]*

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: Should find/develop some metric for comparing how well global structure is preserved?]

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, with small batch sizes direct computation of the t-SNE gradient outperforms the use of auxiliary data structures such as the Barnes-Hut approximation introduced in [26]. While direct computation of the gradient is asymptotically bounded by $\mathcal{O}(k^2)$, where k is the chosen batch size, the operations needed to be carried out for its computation can be efficiently performed in parallel. Choosing a k for which $k \ll n$ and the smaller batch size retains adequate information about the underlying data distribution is at the heart of the considerable runtime advantage our method has over those under comparison.

To demonstrate the real world performance of our proposed method the wall clock times taken by each algorithm to produce embeddings for subsets of the Cytometry B data set have been plotted in figure [add the figure]. [verify the next sentence (preliminary results do show good performance)] The empirical results presented in the figure show

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
VAE	0.9762	0.9247	52037
PCA	0.8557	0.4886	96
UMAP	0.9688	0.9297	74716
t-SNE	0.9879	0.9536	1192812

significant performance gains, even for smaller data set sizes, over the other algorithms under consideration.

- Even with the Barnes-Hut approximation t-SNE is far less scalable in practice. This occurs when a large number of points are in close proximity and the interactions can't be approximated efficiently.
- Optimizing UMAP can be done via approximating stochastically its gradient via edge sampling, this is the key to its performance. Inspired by [37], where dimensionality reduction is done by laying out a graph with the SNE P_{ij} matrix as its adjacency matrix.
-

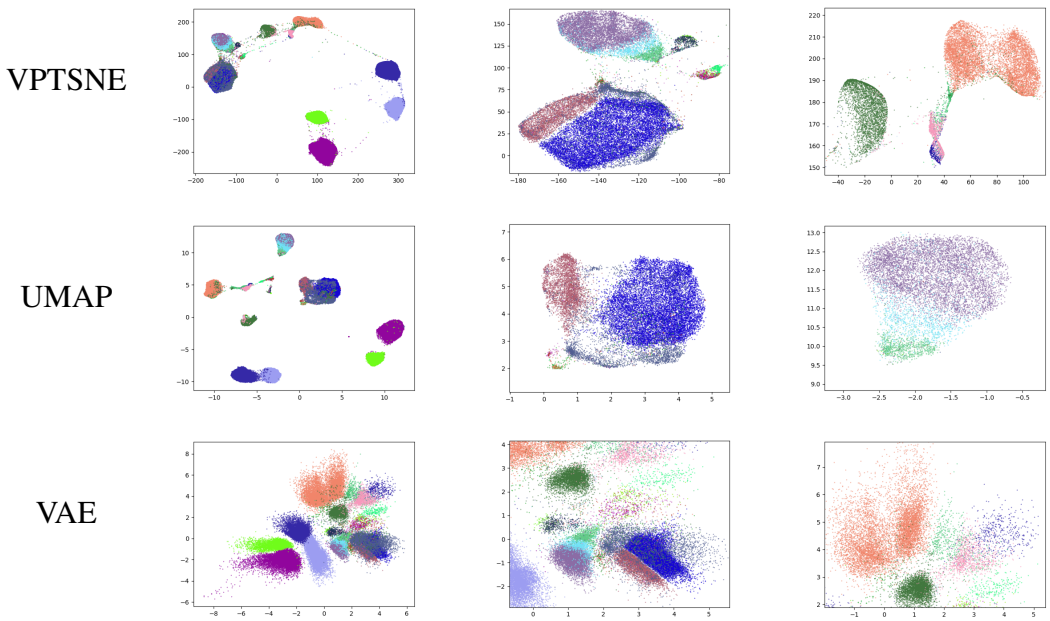

Figure 4.4. *Cytometry A data set embedded with the methods being compared in section 4.5. [Note: Add the rest, currently missing PTSNE, PCA, t-SNE.]*

4.6 Robustness to Sparse Data

As discussed in section [ref. the correct section] 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 random 0.01% subsets of the MNIST data set. As the size of the subsets were only 600 data points each, we were able to use batch gradient descent while training the PTSNE mapping. For training VPTSNE we took advantage of the probabilistic model by sampling data points from the posterior distributions of the training data points, effectively creating different batches of 600 data points at each iteration. Evaluation of the mappings was carried out on the full MNIST test set as in previous experiments. The experiment was repeated 20 times on different subsets. Boxplots corresponding to the obtained scores can be found in figure 4.5. Significant differences in the obtained scores are noted in favor of our method (one-tailed t-test: 1-NN $p = 6.1309E-08$, trustworthiness $p = 9.9365E-05$). 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.

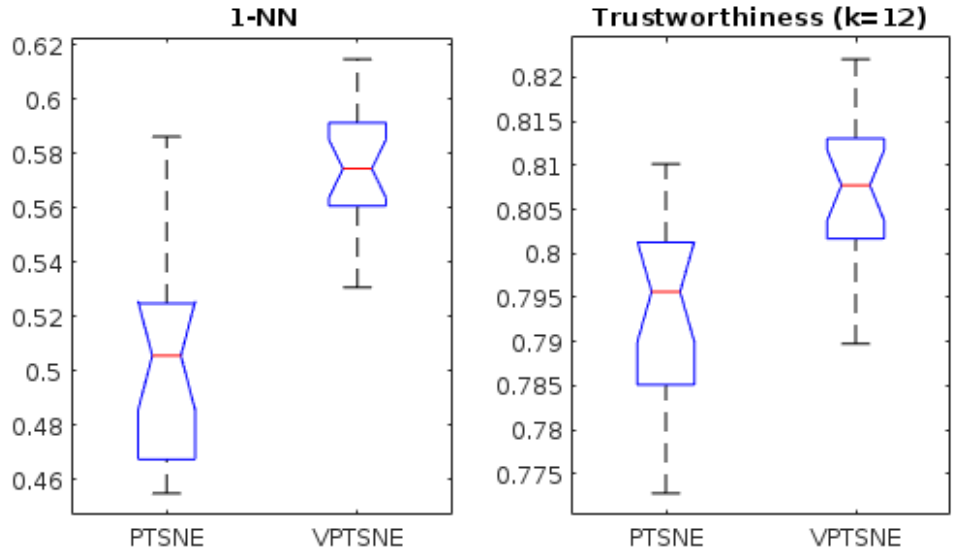

Figure 4.5. *Boxplots of 1-NN and trustworthiness scores obtained for PTSNE and VPTSNE trained on 0.01% subsets of the MNIST data set.*

A plot with sparsity (downsampling factor on x, and goodness on y, like we had previously

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 artefacts. 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 [5, 6] shown in [7]]) 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.

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 [41], where a predetermined fraction of randomly chosen elements of a data point are set to 0. We test robustness by applying masking noise to 20% and 30% fractions of the training set. Following training we test generalization on the uncorrupted test set.

The results of our experiments (see figure 4.6) 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 [18]?]

[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.]

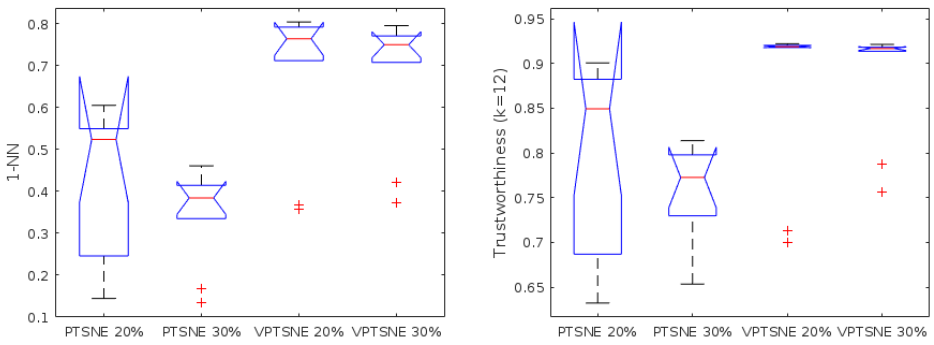

Figure 4.6. *Boxplots of 1-NN and trustworthiness scores obtained for PTSNE and VPTSNE trained on MNIST data with different levels of masking noise applied. Change to line plot with confidence intervals once more test runs are completed for different corruption %.*

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

4.8 Sampling from Hidden Layers

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

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 $d_{hidden} \ll d_{input}$ 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 [25]. We have additionally included the embedding of data points obtained with the original implementation of parametric t-SNE for comparison in figure 4.7. Qualitatively and quantitatively ([still WIP]) 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 the KNN approximation introduced in [26] 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.

[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 $p(\mathbf{x}^{(i)})$ for each data point $\mathbf{x}^{(i)}$. An approximation can be retrieved by importance sampling [34]:

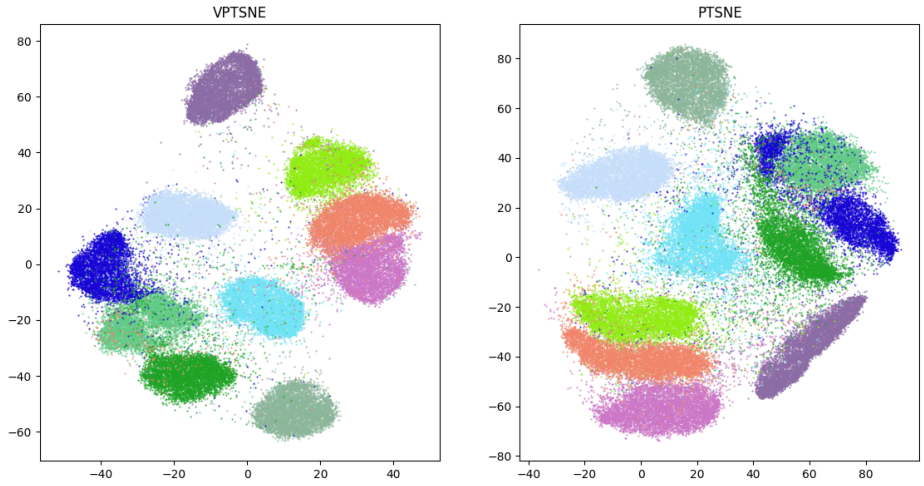

Figure 4.7. *Embedding produced by training on a 7 dimensional latent representation of the MNIST data set contrasted with the original implementation of [25] run on the original data.*

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

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 [29, 42]?]

[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 $\log p(\mathbf{x}) \leq -150$ to separate between the two. In figure 4.8 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.

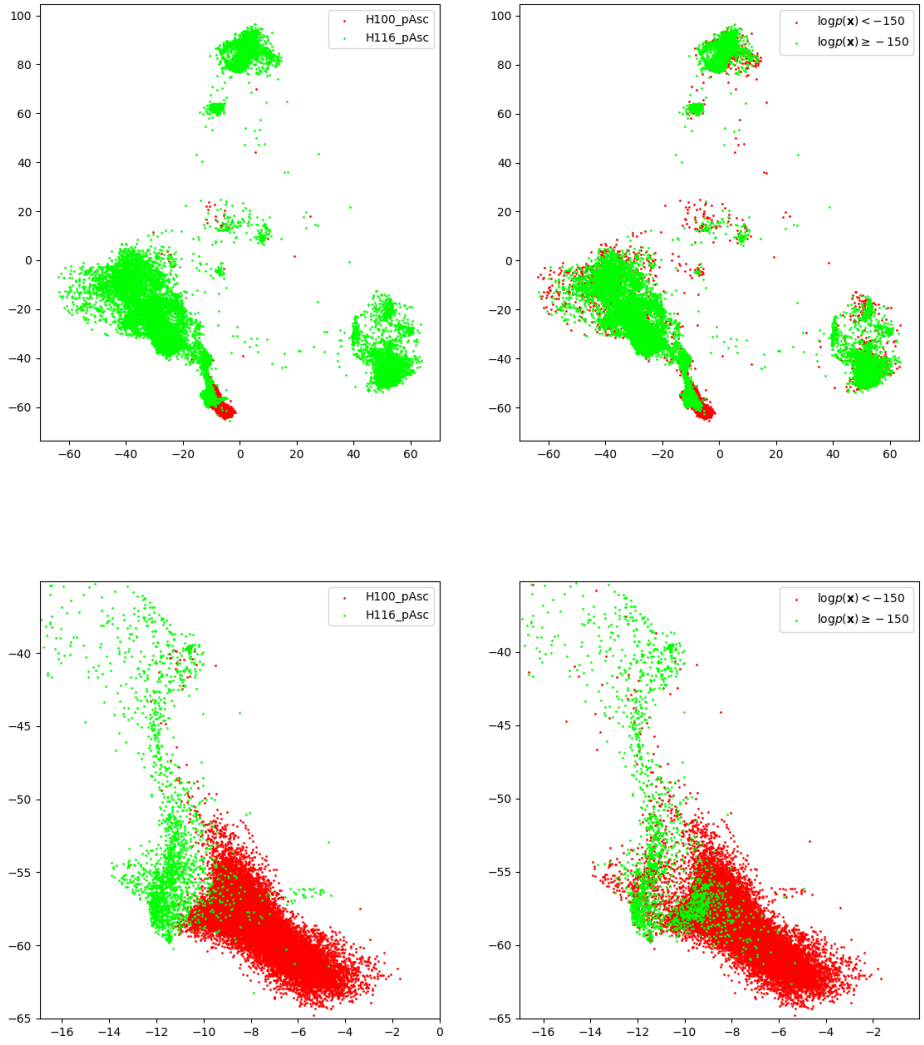

Figure 4.8. *Detecting outliers in cytometry data.* *[Rename patients to match text.]*

5. CONCLUSIONS

[19, 33, 12, 25, 27, 34, 22, 10, 3, 24, 4, 31, 38, 21, 40, 28, 23, 14, 13, **?**, **?**, 36, 15, 35, 44, 26, 39, 16]

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] 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.

[6] 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.

[7] B. Dai, Y. Wang, J. Aston, G. Hua, D.P. Wipf, Veiled attributes of the variational autoencoder, CoRR, Vol. abs/1706.05148, 2017.

[8] 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.

[9] 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.

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

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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

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

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

[18] 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.

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

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

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

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

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

[24] 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.

[25] 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.

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

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

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

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

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

[31] 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.

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

[33] 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.

[34] 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.

[35] 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.

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

[37] 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.

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

[39] 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.

[40] J. Venna, S. Kaski, Visualizing gene interaction graphs with local multidimensional scaling, in: Proceedings of ESANN'06, 14th European Symposium on Artificial Neural Networks, Evere, Belgium, 2006, d-side, pp. 557–562. Poster.

[41] 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.

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

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

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