Exploring UMAP & t-SNE Performance
Project Report for COMS4774

Raghav Mecheri (rm3614) and Ketan Jog (kj2473)

December 2020

Abstract

The ability of stochastic non-linear dimensionality reduction algorithms to "converge" to a fixed embedding as they continue sampling from an underlying fixed data distribution has not been rigorously empirically tested based on our literature review. Furthermore no significant theoretical guarantees have been established for the same. In this project we explore the performance of t-SNE and UMAP. We look at the *quality* and *stability* of embeddings produced by these algorithms as we increase the amount of data they are allowed to use to build a map from the ambient data sampling space to R^2. We further make some interesting observations that might help develop a better intuition in establishing theoretical rigor on the performance of these algorithms. We find that UMAP converges much quicker to a relatively constant lower dimensional representation of the data than t-SNE. We also find that as we sample more data from the underlying distribution, UMAP clusters neighbors together more tightly than t-SNE. We also introduce a concept we call "algorithm-sufficient" data - the size of a sample needed for a dimensionality reduction algorithm to accurately map the original data distribution into the embedding space.

Background: Dimensionality Reduction

This section consists of a brief overview on dimensionality reduction in general, as well as a more in-depth review of t-SNE and UMAP.

Dimensionality Reduction

Dimensionality reduction is the transformation of data from a higher dimensional space into a low dimensional (typically R^2 or R^3) space so that the low dimensional representation retains some meaningful properties of the original data. Some popular dimensionality reduction methods include:

- Principal Component Analysis (PCA): PCA is a linear dimensionality reduction method, based on converting the data's original features into a new feature-set such that the data is projected along the direction of maximum variance.

- Kernelized Principal Component Analysis (kPCA): kPCA involves using kernelization in order to project data into a higher dimensional space, before running PCA on the same. This is a non-linear dimensionality reduction methodology.

- Isomap: Isomap is a non-linear dimensionality reduction method, which is capable of producing an isometric low-dimensional embedding of a higher dimensional space.

- Multidimensional Scaling (MDS): Multidimensional scaling is another methodology, capable of taking an input in the form of a distance matrix, and using the same to create a configuation of points mapped into a lower-dimensional space.

- Other dimensionality reduction methods may include Locally Linear Embedding (LLE), Laplacian Eigenmaps, Nash Embeddings, Variational Autoencoders, etc.

Two extremely popular dimensionality reduction methods used today are t-Stochastic Neighbour Embeddings (t-SNE) and Uniform Manifold Approximation and Projection (UMAP). These are the focus of our experiments and report, and have been explained in further detail below.

t-SNE

Overview

t-SNE is a dimensionality reduction algorithm based, which is an improvement on the Stochastic Neighbor Embedding algorithm. t-SNE is a non-linear method that is primarily used for visualisation, and works towards preserving the local structure for a given dataset.

Optimisation

t-SNE algorithm consists of two parts. First, t-SNE constructs a probability distribution over pairs of high-dimensional objects such that pairs of similar objects are assigned a higher probability, while dissimilar pairs are assigned a lower probability. Once this has been computed, t-SNE defines a similar probability distribution over the points in the low-dimensional map, and it works to minimise the Kullback–Leibler divergence between the two distributions, with respect to the locations of the points being mapped.

Objective Function

$$min_{(i,j)\in X}\ \mathrm{KL}\left(P \parallel Q\right) = \sum_{i\neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{1}$$

UMAP

Overview

Uniform Manifold Approximation and Projection (UMAP) is a dimension reduction technique that can be used for visualisation similarly to t-SNE, but also for general non-linear dimension reduction. UMAP works to preserve both local, and global structure. The algorithm is founded on the below assumptions about the data:

- Our data lies on a Riemannian manifold, where a Riemann manifold is a type of real and smooth manifold which is equipped with a positive-definite inner product q_p on the tangent space T_pM at each point p.
- This manifold is locally connected
- The data is distributed uniformly on the manifold

For a given dataset, UMAP works to model this manifold using a fuzzy topological structure (more below), and then the target embedding is found by searching for a low dimensional graph of the data that has similar topological representation. This is accomplished by minimising a form of cross-entropy loss between these graph structures.

Constructing a topological representation of data

The theory behind UMAP allows us to build a neighborhood graph that leverages topological machinery and establish guarantees on the validity of such a graph being a good representation of the underlying topological space.

Each point is given its own distance function, and then these metrics are made compatible with each other using fuzzy unions of their point unit neighborhoods. A unit ball around a single point is stretched to include its K nearest neighbors. This allows us to build a graph representation of the data in the original space. Our model for the same in the target space is simple, where the edge weight function is simply the desired distance metric.

Loss Function

Using the constructed graph as an original space representation, UMAP optimises for an embedding space representation using cross entropy. The loss function used is as follows:

$$\Sigma_{e \in E} w_h(e) log(\frac{w_h(e)}{w_l(e)}) + (1 - w_h(e)) log(\frac{1 - w_h(e)}{1 - w_l(e)}) \tag{2}$$

Summary

- UMAP enables efficient, non linear dimensionality reduction with a Big O cost of $O(nd)$
- Since UMAP constructs a graph representation for our data set, it is not restricted to Euclidean space. Any metric space is compatible with UMAP.
- UMAP works to preserve both the local, and global structure of a given dataset
- UMAP creates a mapping from R^D to R^d, which allows us to embed new points in existing embedded spaces

Experimental Design

Background

We want to understand how the mapping into R^2 of a known data distribution in a higher dimensional space changes as we allow the algorithm trying to fit this embedding to sample more data. Lets call the embedding generated by the algorithm using t^{th} fraction of the dataset E_t. We wanted to measure the "quality" of E_t as t increased. For this we decided to look at K-nearest neighbor precision, and K-farthest neighbor precision. We wanted to see for higher values of t if UMAP made a trade-off between preserving local structure. We used t-SNE as a control since its loss only preserves local geometry. We define KN Precision and FN Precision as follows (Verma et al).

$$NN - Precision(K) = \frac{1}{NK} \sum_{i=1}^{n} NN_k(y_i) \cap NN_k(x_i) \tag{3}$$

$$FN - Precision(K) = \frac{1}{NK} \sum_{i=1}^{n} FN_k(y_i) \cap FN_k(x_i) \tag{4}$$

We then followed the intuition that if local structure is well preserved, then the nearest neighbours in the original data space should match the nearest neighbours in the embedded space. We define the neighbors $N_\epsilon(x_i), N_\epsilon(y_i)$ of points x_i, y_i by thresholding the pairwise similarity $p_j|i, q_j|i$ respectively, at a selected threshold ϵ (Verma et al).

For a given ϵ similarity threshold, we define the precision as follows:

$$Precision = \frac{N(E_t) \cap N(X)}{N(E_t)} \tag{5}$$

We decided to explore whether the ratio between inter-cluster and intra-cluster distance of the embedded points changed as we increased our data fraction t. To measure this we adopted the LDA score. Our hypothesis was that after a certain amount of data used, the clusters themselves would stabilise for UMAP considering that it has both "push" and "pull" forces in its loss while t-SNE would not stabilise the LDA score as t increased since it disregards local structure.

$$L = \frac{\Sigma_j S_j^2}{\Sigma_j (\mu_j - \mu)(\mu_j - \mu)^T} \tag{6}$$

Then we decided to shift our focus onto the *embedded space*. Suppose we have a sequence of embeddings $\{E_t\}$ produced by an algorithm. Now let f_t be the corresponding maps ($f : R^D \to R^2$) that create these embeddings. For a new sample of data $\{x_i\}_{i=1}^n$ from the underlying distribution X, does f_t approach f for some true f as t increases?

$$f_t(\{x_i\}_{i=1}^n) \to f(\{x_i\}_{i=1}^n) \tag{7}$$

To empirically test this we calculated the average point distance of the embedded sample points between embeddings, and looked at whether this distance diminished/increased or stayed constant.

The Average Point Distance between 2 embeddings E_t and E_k of the original dataset X:

$$APD(E_t, E_k) = \frac{1}{|X|} \sum ||f_t(x) - f_k(x)||^2 \tag{8}$$

In order to try and understand the above behavior, t-SNE's and UMAP's behavior were observed over on three datasets: the MNIST dataset, the Fashion MNIST dataset, and the Olivetti faces dataset. Some notes on our experimental set-up may also be found below, for reference.

Experimental Notes

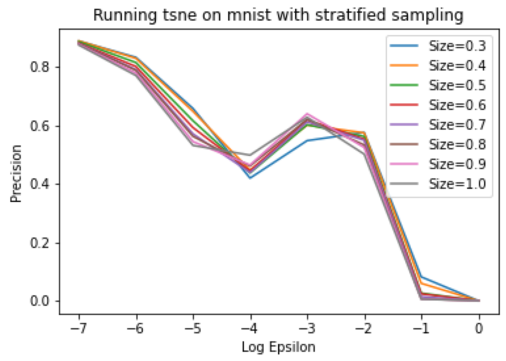
In this section we felt it would be useful to include a few important notes about our experimental setup, in order to provide context about a few issues that we had to tackle through the course of this experimental procedure.

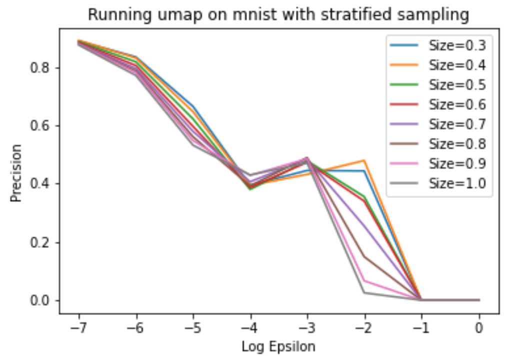
- 10 runs of every experimental combination were recorded and the values averaged, in order to remove potential variance
- Different random seeds were fixed per experimental run, in order to keep experimental results constant within runs
- Embedding new points into an existing t-SNE embedding, while not supported by t-SNE's initial design, was treated as an optimisation problem (Policar et al).

Experimental Inference

As we embed more data UMAP builds tighter clusters than t-SNE

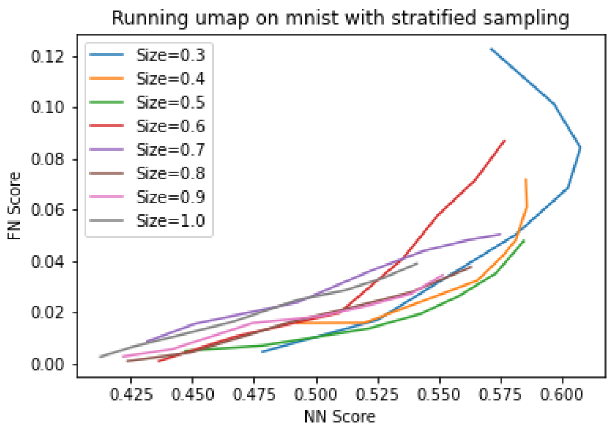
We observed this phenomenon on the MNIST dataset. Epsilon is our similarity threshold. For higher values of epsilon, our ϵ-neighborhood around a given point x would be tighter. We measured the ϵ-neighborhood precision with (Y) the point in the embedded space considered as the "predicted" value. We plotted this precision against its corresponding epsilon value. For lower similarity thresholds we didn't notice any effect of the amount of data on the precision. However for higher values of epsilon (≈ 0.01) we observed that higher values of t resulted in a much lower precision value. Looking at the expression of ϵ-precision we can infer that the size $|N_\epsilon(y_i)|$ increases as we get more data relative to $|N_\epsilon(x_i)|$, meaning that close points are pulled even closer together in UMAP. We contrast this with t-SNE which shows no discernible difference with respect to the dataset fraction t.





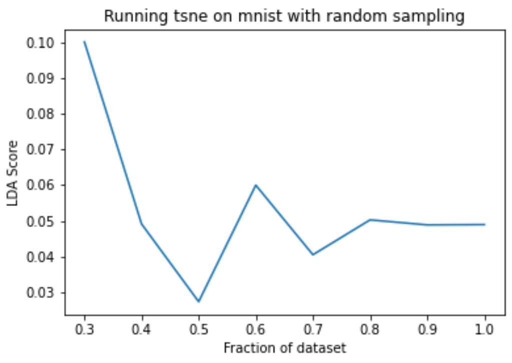
K-Nearest and F-Nearest precision are directly proportional in both UMAP and t-SNE

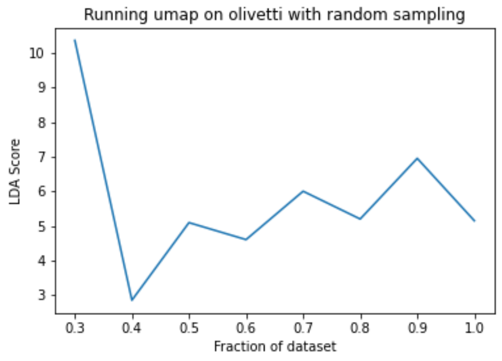
We observed calculated the k-nearest neighbor precision along with the k-farthest neighbor precision for Olivetti and MNIST datasets. We didn't observe the effect of sample size on these metrics for t-SNE or for UMAP. We did observe that the FN precision increased along with NN precision linearly. Due to lack of resources we were not able to verify this, but there is a chance that for larger t, UMAP better preserves global structure. We observed for example on the MNIST dataset that the precision lines with steeper slopes were associated with E_t for large t. This would need further work to establish.



Some datasets-algorithm pairs converge to a stable LDA score

We observed how the LDA score changed for E_t as t increased. We noted that for the Olivetti dataset, the variance of the LDA score across different t was lower than that on MNIST. Furthermore we noticed that for certain algorithm dataset pairs (specifically MNIST with t-SNE and Olivetti with UMAP), after a certain value of t, the LDA score fluctuated within a small interval. We don't know whether this means that this t^{th} fraction is enough to model the embedding of the data distribution in R^2.

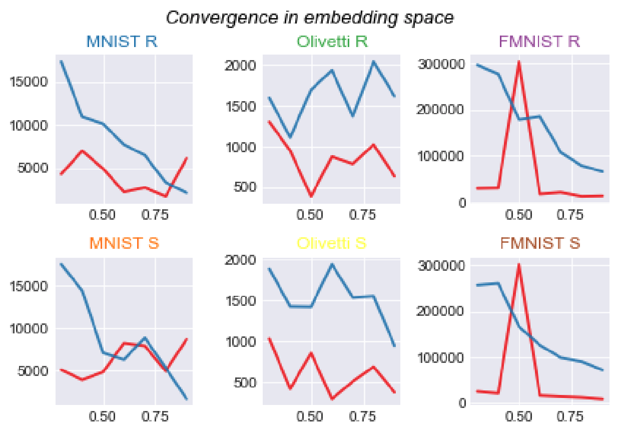


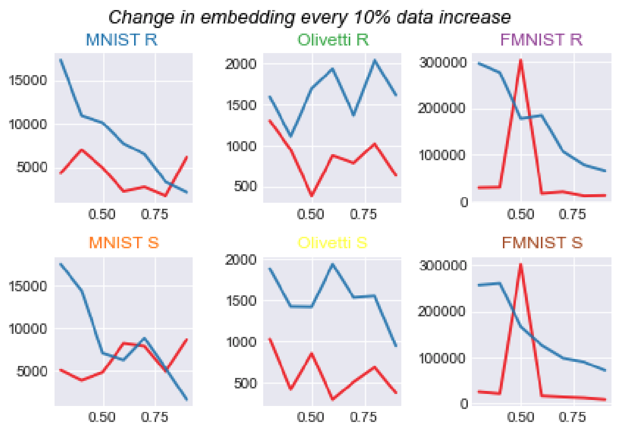


UMAP achieves a stable embedding with much smaller samples from the data distribution

We calculated the average inter-point distances between consecutive E_t and E_{t+1}, to see how much the points deviated from their embedded positions on average as the data fraction increased. We observed that these average deviations for were within a fixed interval for all values of t. That means, the positions of the embedded points in the holdout set as mapped using 30% of MNIST data, was just as close to the ”true position” (ie based on a mapping using all the data) as the positions obtained using 90% of the data. We saw very similar trends on the Olivetti faces dataset as well as the Fashion-MNIST dataset. Interestingly t-SNE showed a very robust trend of moving closer and closer to the ”true position” of the embedded points as we used larger fractions of data. Thus we observed the performance of t-SNE scale directly with the amount of available data. We need further clustering performance analysis to see whether these observations directly imply the existence of a better more separable clustering.

In the following figures, tsne is the blue line, UMAP is the red line. The x-axis shows the fraction of data used, while the y-axis shows the APD value. Tile-titles ending in ’R’ imply the data was randomly sampled, while those ending in ’S’ mean we performed stratified sampling.





Method of sampling had no discernible effect on results

Every experiment we ran was performed under uniform stratified sampling as well as under random sampling. We would like to note the the results do not show a discernible pattern of effect privy to either of the sampling methods. This is observed in the figures shown above.

Future Work

We're extremely encouraged by our initial results, and while we definitely believe that these are preliminary conclusions, we also do strongly believe that this project can be built on, in order to come to some interesting conclusions.

Studying distinct data configurations

1. Can we understand which data configurations scale better with size? Our current experiments show us a trend: certain datasets seem to scale better with size, but we could be mistaken here
2. It would also be interesting to investigate, post some empirical results, whether there could be a mathematical formulation to quantify how well a dataset scales with size, in the context of performance during dimensionality reduction tasks.

Studying the influence of the applied metric

Newer versions of t-SNE allow for different, non-euclidean metrics to also be used during the dimensionality reduction process, while UMAP allows us to set different $(input, output)$ metric pairs for a given task. We feel that it may be interesting to study how different metrics affect the tendency for these algorithms to converge as the data size increases

Convergence across dimensions

1. A lot of the techniques that have been used to evaluate the performance of our algorithms depend on dimensionality, either directly or indirectly.
2. We think it could be interesting to try and better understand a methodology to measure performance across different high-dimensional spaces, and work towards doing so without allowing dimensionality to contort our results.

Clustering performance using UMAP

We think it could be interesting to study various clustering algorithms on high dimensional data, after using UMAP as a pre-processing step. Specifically, we feel that it may be interesting to understand how the stability of the clustering is affected by increasing the value of n. It also may be interesting to understand if there exists a sufficient amount of data for a given algorithm, that is capable of modelling a certain type of distribution well.

Semi Supervised UMAP

We feel that it may be interesting to explore how UMAP's performance is improved by labelling a small subset of the data we're attempting to embed into a target space, in order to understand UMAP's potential in semi-supervised learning tasks.

Works Cited

1. Our experimental framework and its source code may be found at:
`https://github.com/raghavmecheri/DimConvergence`

2. Convergence and Rate of Convergence of A Manifold-Based Dimension Reduction Algorithm:
`https://papers.nips.cc/paper/2008/file/735143e9ff8c47def504f1ba0442df98-Paper.pdf`

3. Scikit-learn: Machine Learning in Python:
`https://scikit-learn.org/stable/index.html`

4. openTSNE: a modular Python library for t-SNE dimensionality reduction and embedding:
`https://www.biorxiv.org/content/10.1101/731877v3`

5. Stochastic Neighbor Embedding under f-divergences:
`https://arxiv.org/pdf/1811.01247.pdf`

6. An Analysis of the t-SNE Algorithm for Data Visualization:
`http://proceedings.mlr.press/v75/arora18a/arora18a.pdf`

7. UMAP dimensionality reduction improves clustering performance:
`https://link.springer.com/chapter/10.1007/978-3-030-51935-3_34`

8. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction:
`https://arxiv.org/abs/1802.03426`

9. Visualizing Data using t-SNE:
`https://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf`

10. Learning about Dimensionality Reduction and Topological Data Analysis:
`ttp://www.cs.columbia.edu/~verma/classes/uml/index.html`

11. Understanding f-divergence:
`http://people.lids.mit.edu/yp/homepage/data/LN_fdiv.pdf`