MiniProject 4: Reproducibility in ML

Harms Simon, Madrigal Ariel, and Quinsey Michael

1 Abstract

In this project, we have reproduced some results from the UMAP paper (McInnes et al. [2018]): "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction". In our reproduction, we observed results supporting the claims made in the original paper. Examining each claim made in the paper and trying to recreate it, we experimented with various different hyperparameters in UMAP and were able to reproduce most of the results.

2 Introduction

The UMAP technique involves distributing the data on a locally connected Riemannian manifold with a locally constant Riemannian metric. With this distribution, the data can be modelled and visualized in an approximate topological structure. It can be useful for visualizing large clusters of data points and their relative proximities.

It was created as an improvement upon the t-SNE technique, preserving both global and local structure and operating at a higher speed.

3 Scope of reproducibility

The overall goal of the UMAP paper is to learn the manifold of the data and perform dimension reduction. The authors claim to offer a similar performance in terms of visualization quality to t-SNE, while providing a better global structure of the data.

The first results that the authors provide are the effect of varying the hyperparameters min-dist and number of neighbors across various datasets. The claim from the authors is that UMAP can distinguish both fine grained and large scale manifold features by controlling the min dist and number of neighbors parameters. These results are shown in the original paper in Figures 1-3. We reproduced these experiments in our first claim.

Then, the authors make a qualitative comparison of the embedding produced by different algorithms, including PCA (Hotelling [1933]), t-SNE (Van der Maaten and Hinton [2008]) , LargeVis (Tang et al. [2016]), Laplacian Eigenmaps (Belkin and Niyogi [2003]) and UMAP. The claim here is that UMAP can produce a better embedding than other methods in terms of recapitulating the local and global structure of the data. These results are displayed in Figure 4 from the original paper. We reproduced these experiments in our second claim.

After that, the authors make a quantitative assessment of the multiple algorithms mentioned above, measuring the kNN classifier accuracy on each method embedding. In here, the claim is that UMAP offers the best performance in terms of local manifold preservation. These results are displayed in Figures 5-6 and Table 1-2 from the original paper. We reproduced these experiments in our third claim.

Finally, we analyze two hyperparameters that are not discussed in the UMAP paper, a and b and the effect on the embedding produced by UMAP. These experiments consist our fourth claim.

- Claim 1. UMAP can recapitulate both local and global manifold features by modifying the hyperparameters min-dist and number of neighbors.
- Claim 2. UMAP offers a better dimension reduction embedding than other algorithms.
- Claim 3. UMAP offers better performance at recapitulating local structure than other methods.
- Claim 4. UMAP can recapitulate global manifold structure by modifying the hyperparameters a and b.

Submitted to ML Reproducibility Challenge 2020. Do not distribute.

4 Methodology

We used the author's code or previous generated libraries for the implementation of the algorithms. For umap, we used the `umap-learn library`, which is original implementation from the authors. For PCA, we loaded the library from `sklearn.decomposition`. For t-SNE, we loaded the library from `sklearn.manifold`. For LaplacianEigenmaps, we installed the library `fastlapmap`. We also used the KNN Classifier from `sklearn.neighbors`.

4.1 Model descriptions

The UMAP algorithm can be described in two steps. In the first step, the UMAP algorithm constructs a fuzzy topological representation. This is usually done by computing a weighted k-neighbour graph. In the second step, the algorithm finds a low dimensional representation by optimizing a representation that is as close to the fuzzy topological representation derived in the first step. This step is performed with stochastic gradient descent. The algorithm learns an embedding of dimensions specified by the user as an hyperparameter. In this project, we trained the model for each dataset, so we didn't used any pre-trained model.

4.2 Datasets

We tested the technique on the COIL-20 (Rate and Retrieval [2011]), PenDigits (Alpaydin and Alimoglu [1998]), and MNIST (LeCun [1998]) data sets, which are three of the data sets used in the original paper. The COIL-20 data set contains 1440 128x128 greyscale images of 20 objects rotated 72 different ways. The PenDigits database consists of 1797 8x8 greyscale images. The MNIST database contains 70000 28x28 grey-scale images of hand drawn digits. No preprocessing was done for any of the datasets. In the original paper, other datasets were included, but were omitted in this reproducibility report due to higher computational requirements needed in terms of memory and runtime.

4.3 Hyperparameters

The UMAP algorithm has 4 four main hyperparameters that are discussed in the article: 1)number of neighbors, 2) the number of dimensions for the resulting embedding, 3) min-dist, the desired separation between close points in the embedding space and 4) n-epochs, the number of training epochs to use when optimizing the low dimensional representation. As part of the original article, an hyperparameter search was made for the two most important hyperparameters: number of neighbors and min-dist. This was done by qualitatively inspection of the resulting embeddings as part of our first claim. We found that values in the range of min dist 0.05 - 0.2 and number of neighbors 5-20 were adequate for the datasets that we inspected. We also inspected the effect of the hyperparameters a and b, which have an effect on the min dist selection (Melville [2022]). These results are explored in Claim 4.

4.4 Experimental setup

For Claim1, we evaluated qualitatively the effect of varying min-dist and number of neighbors in the PenDigits and MNIST datasets. For Claim2, we evaluated qualitatively the embedding produced by different algorithms in the PenDigits and COIL-20 datasets. For Claim3, we compared the local structure recapitulated by each algorithm with the kNN accuracy in the PenDigits and COIL-20 datasets. To estimate the accuracy, we performed a 10-fold cross validation. Finally, for Claim4, we also evaluated qualitatively the effect of varying a and b in the PenDigits dataset.

4.5 Computational requirements

The computational resources used for this report were based on the standard Google Colab allocation, which was 12.68GB RAM and 107 GB Disk. For the experiments that involved the PenDigits and COIL-20 datasets, the runtime was less than 1 hour in the Google Colab CPU. For the experiments that involved the MNIST dataset, the runtime was less than 2 hours in the Google Colab CPU.

5 Results

We believe that our results support the main claims of the original paper. In the next section, we describe each of our results and how they align with the original article.

5.1 Results reproducing original paper

5.1.1 Result 1

This result supports claim 1. We varied the hyperparameters min dist and number of neighbors for the PenDigits dataset (Fig. 1) and the MNIST dataset (Fig. 2). We observed that low values of number of neighbors favored a better representation of the local structure, while larger values favored the global structure. In the MNIST dataset we are able to see that the global structure is preserved, meaning that similar digits (e.g. 3,5,8) are clustered together, while more distinct digits (e.g. 0) are clustered apart. For min dist, larger values allowed a better distinction of individual samples in the embedding but with the cost of losing the fine structure of the data. Altogether, we successfully reproduced the associated experiment from the original paper.

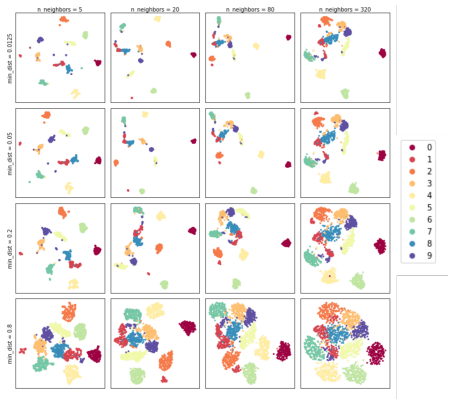

Figure 1: Variation of UMAP hyperparameters n and min-dist for PenDigits dataset.

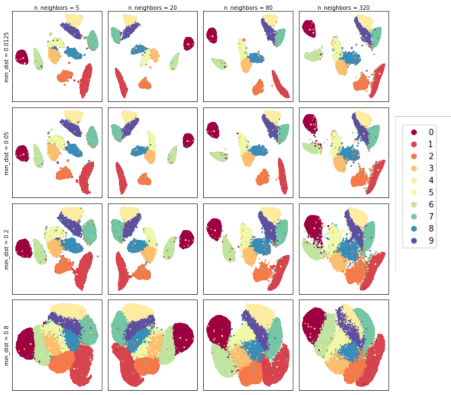

Figure 2: Variation of UMAP hyperparameters n and min-dist for MNIST dataset.

5.1.2 Result 2

This result supports claim 2. We compared the embedding results of PCA, Laplacian Eigenmaps, t-SNE and UMAP in the COIL-20 (Fig. 3) and PenDigits (Fig. 4) datasets. Qualitatively, t-SNE and UMAP offered similar embeddings, they preserved better the local structure of the data (separating different clusters) compared to PCA and Laplacian Eigenmaps. In terms of global structure of the data, there seems to be a subtle difference between UMAP and t-SNE. UMAP seems to preserve better some aspects of the global structure, for instance, it can recover more loops in the COIL-20 dataset, including the intertwined loops. Collectively, our results recapitulate the main findings from the original paper.

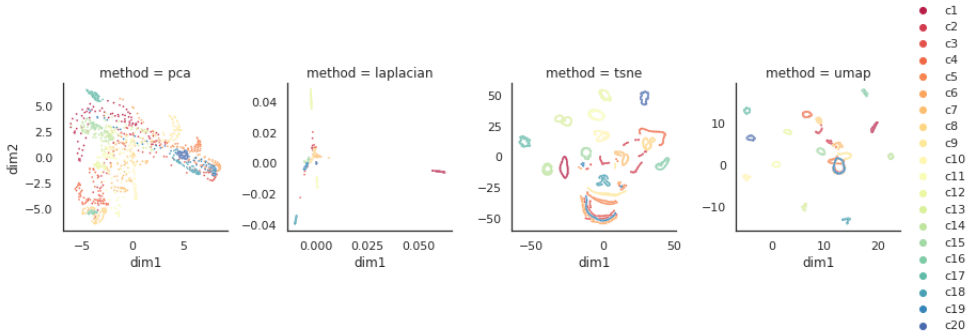

Figure 3: Comparison of dimension reduction algorithms for COIL-20 dataset.

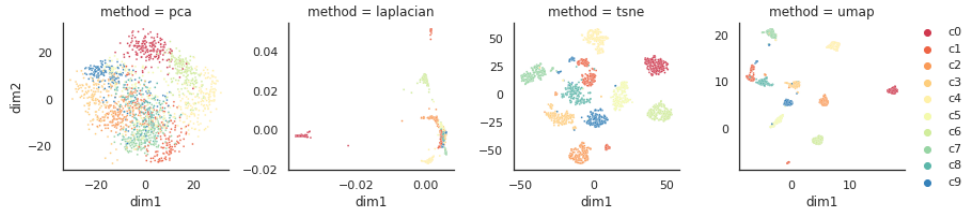

Figure 4: Comparison of dimension reduction algorithms for PenDigits dataset.

5.1.3 Result 3

This result supports claim 3. We compared quantitatively the ability of UMAP against other methods at preserving local structure in the data (Fig. 5) (Table 1). To compare, we estimated the kNN classifier accuracy on the embedding produced by each method, using a 10-fold cross-validation in the COIL-20 and PenDigits datasets. Individual accuracies can be found in (Fig. 5) , while mean accuracies in (Table 1). We observed that UMAP and t-SNE offered better performance in comparison to PCA and Laplacian Eigenmaps, and that in the majority of the instances UMAP and t-SNE offered very similar performance. There was a slight difference in mean accuracy between UMAP and t-SNE in the COIL-20 dataset at high K values, however it was not as prominent as it that was shown in Figure 5 from the original paper.

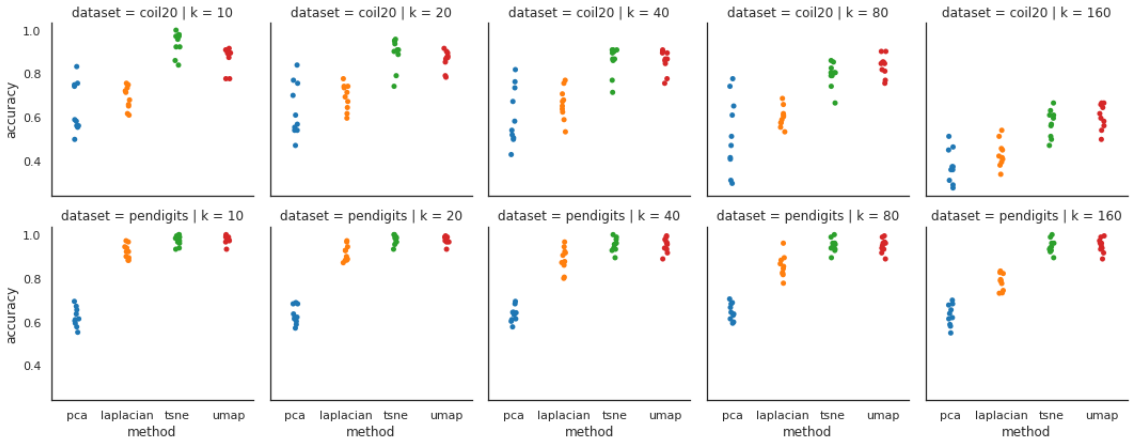

Figure 5: kNN Classifier accuracy estimates (10-fold CV) across various dimension reduction methods and k values, for the COIL-20 and PenDigits datasets.

5.1.4 Additional Result 1

This result supports claim 4. We varied the hyperparameters 'a' and 'b' in the PenDigits dataset (Fig. 6). We observed that while the change of 'a' had low effect on the PenDigits dataset, b seemed to have a drastic effect in the resulting embedding. Low values of b (e.g. 0.25) generated very dense clusters and seemed to favor local structure, even splitting the original clusters of digits. Values higher than 0.5 seemed to generate good visualizations, generating a good tradeoff between preserving the local and global structure of the data. This result in in concordance with an independent analysis (Wang et al. [2021]).

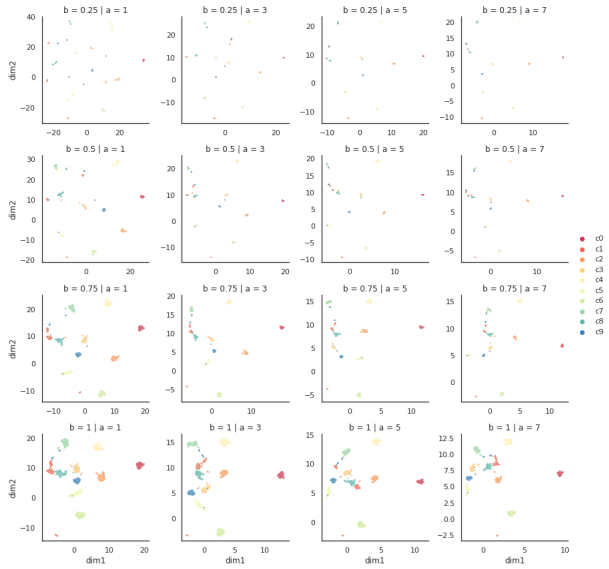

Figure 6: Variation of UMAP hyperparameters a and b in the pendigits dataset.

6 Discussion

Our experimental results support the claims of the paper. We were able to reproduce the better performance against classic methods such as PCA and the better preservation in terms of global structure of the embedding in comparison to t-SNE. We observed some minor differences in terms of KNN performance compared to the original paper, but this could be derived from the noise of a 10-fold cross-validation or the inherent stochasticity of methods such as UMAP and t-SNE. However, we acknowledge that our approach had various weaknesses. First, we were not able to replicate the experiments in the large datasets (Mouse scNRA-seq, Flow cytometry, GoogleNews word vectors) due to limitations in the runtime and available memory from out Google Colab allocation. We recognise that some of the results with larger datasets could have offered a better insight in the effect of some of the hyperparameters of UMAP (e.g. the effect of number of neighbors), as we would expect that higher n would be required for larger datasets. We also were not able to use one of the methods that were used in the comparison (LargeVis), due to difficulties in the runtime. In terms of additional experiments that were not in the original paper, we analyzed the effect of varying the hyperparameters a and b in UMAP. Our results are in concordance with other papers analyzing UMAP hyperparameters (Wang et al. [2021]).

6.1 What was easy

We recognise that the UMAP library in python has an excellent documentation, the code is clear and is formatted in the same way as other sklearn modules, which makes it use very accessible to the community. Also, it had different usage examples on various datasets that allowed us to easily implement the method in our experiments.

6.2 What was difficult

One aspect that we think was difficult to evaluate in the original paper was that the authors argue about the improvement of learning global manifold structure by UMAP compared to t-SNE. In most cases, this was done qualitatively (by

examining cluster distribution in the resulting embedding), and although we agree that this is valid, we also think that perhaps quantitative measures of global structure could offer a clearer perspective on the improvement offered by UMAP. One of the measures that could be applied for future evaluations is the Random Triplet Accuracy (Wang et al. [2021]).

6.3 Communication with original authors

None

7 Statement of Contributions

All authors wrote the report.

References

E Alpaydin and Fevzi Alimoglu. Pen-based recognition of handwritten digits data set. university of california, irvine. *Machine Learning Repository. Irvine: University of California*, 4(2), 1998.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.

Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

James Melville. a and b umap. *https://jlmelville.github.io/uwot/abparams.html*, 2022.

C Rate and C Retrieval. Columbia object image library (coil-20). *Computer*, 2011.

Jian Tang, Jingzhou Liu, Ming Zhang, and Qiaozhu Mei. Visualizing large-scale and high-dimensional data. In *Proceedings of the 25th international conference on world wide web*, pages 287–297, 2016.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9 (11), 2008.

Yingfan Wang, Haiyang Huang, Cynthia Rudin, and Yaron Shaposhnik. Understanding how dimension reduction tools work: An empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization. *Journal of Machine Learning Research*, 22(201):1–73, 2021. URL `http://jmlr.org/papers/v22/20-1061.html`.

7.1 Appendix

Table 1: Mean and standard deviation kNN Classifier accuracy estimates (10-fold CV) across various dimension reduction methods and k values, for the COIL-20 and PenDigits datasets.

dataset	k	pca	laplacian	tsne	umap
COIL-20	10	0.644 (+- 0.109)	0.69 (+- 0.051)	0.941 (+- 0.051)	0.874 (+- 0.05)
COIL-20	20	0.636 (+- 0.116)	0.694 (+- 0.057)	0.894 (+- 0.068)	0.865 (+- 0.042)
COIL-20	40	0.608 (+- 0.125)	0.663 (+- 0.068)	0.861 (+- 0.063)	0.858 (+- 0.05)
COIL-20	80	0.521 (+- 0.161)	0.601 (+- 0.044)	0.795 (+- 0.053)	0.837 (+- 0.046)
COIL-20	160	0.378 (+- 0.073)	0.433 (+- 0.057)	0.574 (+- 0.059)	0.604 (+- 0.055)
PenDigits	10	0.622 (+- 0.041)	0.923 (+- 0.031)	0.974 (+- 0.023)	0.979 (+- 0.018)
PenDigits	20	0.632 (+- 0.039)	0.917 (+- 0.035)	0.977 (+- 0.02)	0.977 (+- 0.017)
PenDigits	40	0.635 (+- 0.034)	0.887 (+- 0.052)	0.953 (+- 0.03)	0.952 (+- 0.031)
PenDigits	80	0.647 (+- 0.036)	0.855 (+- 0.048)	0.951 (+- 0.029)	0.95 (+- 0.03)
PenDigits	160	0.631 (+- 0.046)	0.784 (+- 0.036)	0.947 (+- 0.03)	0.951 (+- 0.031)